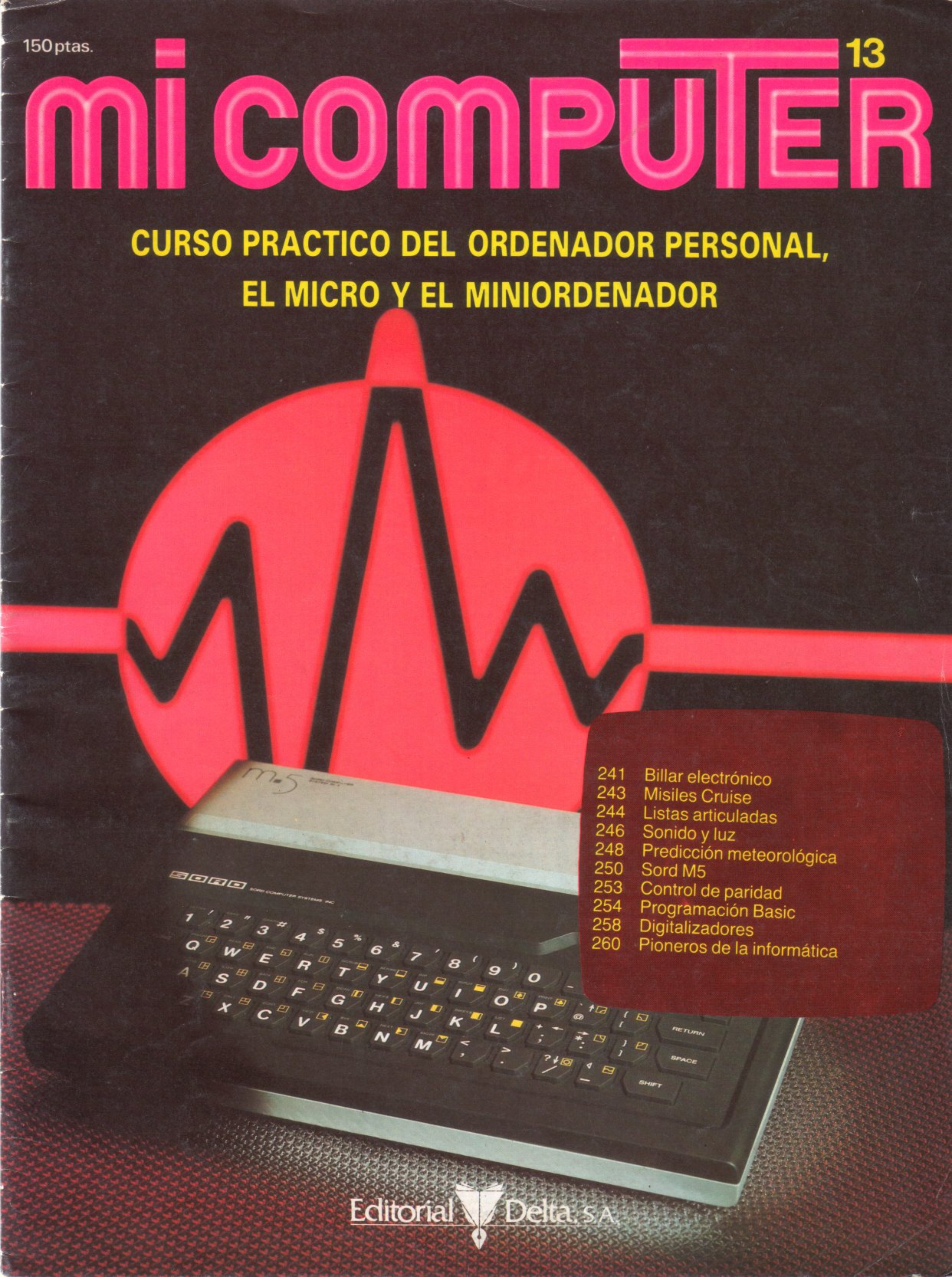

150 ptas.
mi computer
13
CURSO PRACTICO DEL ORDENADOR PERSONAL,
EL MICRO Y EL MINIORDENADOR
241 Billar electrónico
243 Misiles Cruise
244 Listas articuladas
246 Sonido y luz
248 Predicción meteorológica
250 Sord M5
253 Control de paridad
254 Programación Basic
258 Digitalizadores
260 Pioneros de la informática
Editorial Delta, S.A.

MI COMPUTER CURSO PRACTICO
DEL ORDENADOR PERSONAL, EL MICRO Y EL MINIORDENADOR

Publicado por Editorial Delta, S.A., Barcelona, y comercializado en exclusiva por Distribuidora Olimpia, S.A., Barcelona

Volumen II - Fascículo 13

Director: José Mas Godayol
Director editorial: Gerardo Romero
Jefe de redacción: Pablo Parra
Coordinación editorial: Jaime Mardones
Asesor técnico: Jesús Nebra

Redactores y colaboradores: G. Jefferson, R. Ford, S. Tarditti, A. Cuevas
Para la edición inglesa: R. Pawson (editor), D. Tebbutt (consultant editor), C. Cooper (executive editor), D. Whelan (art editor), Bunch Partworks Ltd. (proyecto y realización)

Realización gráfica: Luis F. Balaguer

Redacción y administración:
Paseo de Gracia, 88, 5.°, Barcelona-8
Tels. (93) 215 10 32 / (93) 215 10 50 - Télex 97848 EDLTE

MI COMPUTER, *Curso práctico del ordenador personal, el micro y el miniordenador,* se publica en forma de 96 fascículos de aparición semanal, encuadernables en ocho volúmenes. Cada fascículo consta de 20 páginas interiores y sus correspondientes cubiertas. Con el fascículo que completa cada uno de los volúmenes, se ponen a la venta las tapas para su encuadernación.
El editor se reserva el derecho de modificar el precio de venta del fascículo en el transcurso de la obra, si las circunstancias del mercado así lo exigieran.

ISBN: 84-85822-83-8 (fascículo) 84-85822-90-0 (tomo 2)
84-85822-82-X (obra completa)
Depósito Legal: B. 52/1984

Fotocomposición: Tecfa, S.A., Pedro IV, 160, Barcelona-5
Impresión: Cayfosa, Santa Perpètua de Mogoda (Barcelona) 118404
Impreso en España - Printed in Spain - Abril 1984

Editorial Delta, S.A., garantiza la publicación de todos los fascículos que componen esta obra.

Distribuye para España: Marco Ibérica, Distribución de Ediciones, S.A., Carretera de Irún, km 13,350. Variante de Fuencarral, Madrid-34.
Distribuye para Argentina: Viscontea Distribuidora, S.C.A., La Rioja 1134/56, Buenos Aires.
Distribuye para Colombia: Distribuidoras Unidas, Ltda., Transversal 93, n.° 52-03, Bogotá D.E.
Distribuye para México: Distribuidora Intermex, S.A., Lucio Blanco, n.° 435, Col. San Juan Tlihuaca, Azcapotzalco, 02400, México D.F.
Distribuye para Venezuela: Distribuidora Continental, S.A., Ferrenquín a Cruz de Candelaria, 178, Caracas, y todas sus sucursales en el interior del país.

Pida a su proveedor habitual que le reserve un ejemplar de MI COMPUTER. Comprando su fascículo todas las semanas y en el mismo quiosco o librería, Vd. conseguirá un servicio más rápido, pues nos permite realizar la distribución a los puntos de venta con la mayor precisión.

Servicio de suscripciones y atrasados (sólo para España)

Las condiciones de suscripción a la obra completa (96 fascículos más las tapas, guardas y transferibles para la confección de los 8 volúmenes) son las siguientes:

a) Un pago único anticipado de 16 690 ptas. o bien 8 pagos trimestrales anticipados y consecutivos de 2 087 ptas. (sin gastos de envío).

b) Los pagos pueden hacerse efectivos mediante ingreso en la cuenta 3371872 de la Caja Postal de Ahorros y remitiendo a continuación el resguardo o su fotocopia a Distribuidora Olimpia (Paseo de Gracia, 88, 5.°, Barcelona-8), o también con talón bancario remitido a la misma dirección.

c) Se realizará un envío cada 12 semanas, compuesto de 12 fascículos y las tapas para encuadernarlos.

Los fascículos atrasados pueden adquirirse en el quiosco o librería habitual. También pueden recibirse por correo, con incremento del coste de envío, haciendo llegar su importe a Distribuidora Olimpia, en la forma establecida en el apartado b).

Para cualquier aclaración, telefonear al (93) 215 75 21.

No se efectúan envíos contra reembolso.

Editorial Delta, S.A.

Billar electrónico

El Pinball Construction Set —un avance notable en el diseño de software— permite diseñar juegos en la pantalla del Apple

Incluso en una industria de desarrollo tan rápido como la de los microordenadores, en la cual, razonablemente, cabe esperar que los adelantos de importancia sean casi una cosa corriente, resulta difícil encontrar un producto que sea radicalmente diferente tanto en concepto como en calidad. Un software que reúne estos requisitos es el PCSA *(Pinball Construction Set)*, de la firma Budgeco. Funciona con un Apple II de 48 Kbytes, con una unidad de disco y una palanca de mando. Este paquete realiza una función aparentemente sencilla. Suministra al usuario la figura de una máquina de «millón» (billar electrónico) vacía, y un menú de 38 tipos diferentes de «mobiliario» que sirven para equiparla según desee el propio jugador. Además, existe una serie de funciones de entre las cuales elegir los «útiles» a emplear.

Una vez se haya llenado la mesa de juego según los deseos del usuario —se pueden situar hasta 128 piezas, pero no existe límite en el número de veces que es posible emplear una de cualquier tipo—, ya se puede empezar a jugar. Para ello se selecciona aún otra función con la palanca de mando.

Pueden jugar por turno hasta cuatro personas, pero cada una de ellas sólo puede emplear una bola, en vez de las tres de que se dispone en la mayoría de las máquinas de «millón». Al final del juego, apretando ESCAPE se pueden utilizar otra vez todos los componentes que se desee. Después de cada juego, la forma más interesante de continuar es introduciendo nuevos elementos para hacerlo más complicado.

Por su concepción y realización, el PCS se orienta hacia un software que sea agradable para el usuario. Tan pronto como se carga el programa (lo cual sólo requiere introducir el disco y apretar RETURN), se controla prácticamente toda la acción mediante la palanca de mando. El primer útil a emplear es una mano. Ésta se mueve de forma que apunte a un objeto de los existentes en el menú —que puede ser un parachoques (bumper) o una aleta (flipper)—, y al apretar el botón de la palanca de mando, la mano «recoge» el objeto indicado. Enseguida ésta lo coloca en la posición deseada, y cuando se suelta el botón de la palanca de mando, el objeto queda fijado firmemente en su emplazamiento.

Aquí, lo más interesante es que no sólo se está moviendo la colección de datos que define la forma del objeto, sino también el conjunto de normas que dirigirán la manera de actuar cuando se empiece a jugar. Una aleta, por ejemplo, siempre se mueve 45 grados, primero hacia arriba y luego otra vez hacia abajo. Un parachoques siempre repele la bola, al mismo tiempo que la acelera según un factor de reacción definible. La bola obedece las leyes de Newton sobre movimiento, y cae de acuerdo con la ley de gravedad.

No obstante, además de lo ya enumerado, existe un útil (representado con mucha propiedad por el símbolo de un planeta iluminado parcialmente por el Sol) que permite alterar los parámetros del mundo real: ¡la fuerza de gravedad, por ejemplo, o incluso el tiempo! Esta función es también controlada por la palanca de mando. La posición de cada valor según una escala se altera igual que el mando



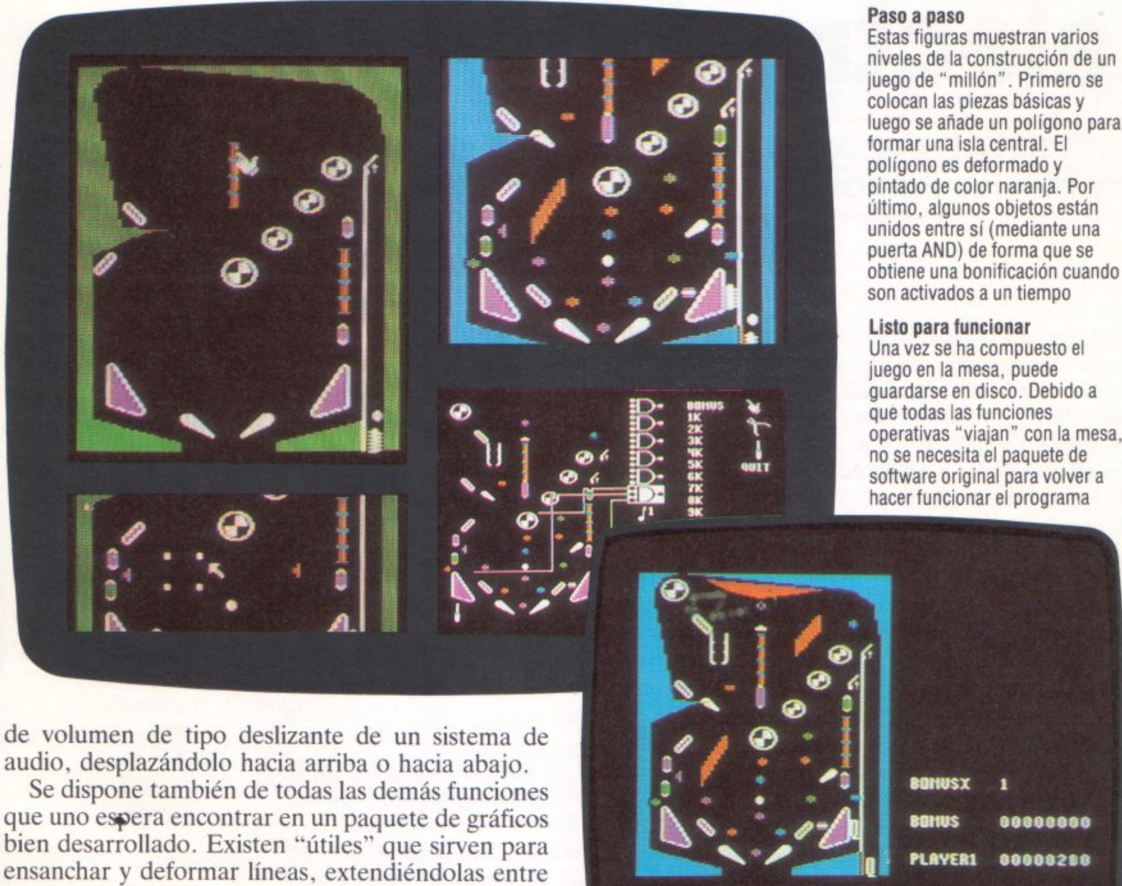
Juegos "hágalo usted mismo"
El PCS (Pinball Construction Set) consta de una mesa vacía, diversos tipos de "mobiliario" (parachoques, blancos, aletas, etc.) y, en la columna de la derecha, los útiles para situar los objetos en la mesa. Esta columna contiene también las funciones para ajustar el tamaño, forma, color y grado de influencia recíproca entre las piezas, así como para guardar en disco los juegos una vez completos

Ian McKinnell



Cosa de niños
El PCS produce incluso sonidos auténticos y el equivalente a luces centelleantes. Pero realmente es más divertido idear y construir los juegos que jugar con ellos. Ahora bien, si tuviera incorporado un mando de FALTA (TILT)...

The Image Maker



Ian McKinnell

Paso a paso
Estas figuras muestran varios niveles de la construcción de un juego de "millón". Primero se colocan las piezas básicas y luego se añade un polígono para formar una isla central. El polígono es deformado y pintado de color naranja. Por último, algunos objetos están unidos entre sí (mediante una puerta AND) de forma que se obtiene una bonificación cuando son activados a un tiempo

Listo para funcionar
Una vez se ha compuesto el juego en la mesa, puede guardarse en disco. Debido a que todas las funciones operativas "viajan" con la mesa, no se necesita el paquete de software original para volver a hacer funcionar el programa

Ian McKinnell



Chris Stevens

Perspectiva objetiva

Además de ser un juego educativo y fascinante, el Pinball Construction Set es un buen ejemplo de programación orientada de objetos. En programación normal, la estructura de los datos está definida, y las rutinas del programa están escritas para manipularlos. En programación orientada de objetos, los cálculos y procedimientos son inseparables de los datos. En el programa que estamos tratando, mover el símbolo de una aleta (flipper) para situarlo en la mesa, no sólo establece el dato (en este caso la forma de la aleta), sino que dispone las rutinas asociadas de forma que sean capaces de activar la aleta.
La programación orientada de objetos se presta para aplicaciones visuales. Las hojas electrónicas constituyen otro ejemplo: el campo que muestre un resultado contendrá también la fórmula para obtener este resultado.
La tendencia actual a la representación de los componentes de un centro de gestión deriva de la misma idea. Indicando en la pantalla la imagen de una hoja de papel en blanco, se activa el procesador de textos, mientras que al hacer referencia a un esquema en miniatura de un gabinete de archivo, se obtiene la clasificación de los resultados

de volumen de tipo deslizante de un sistema de audio, desplazándolo hacia arriba o hacia abajo.

Se dispone también de todas las demás funciones que uno espera encontrar en un paquete de gráficos bien desarrollado. Existen "útiles" que sirven para ensanchar y deformar líneas, extendiéndolas entre nodos predeterminados (llamados *bandas elásticas*); otros se emplean para pintar los bloques con uno de los colores de la paleta y para ampliar zonas pequeñas de la imagen.

Sin embargo, la característica más importante del PCS no reside en sus funciones individuales y su versátil capacidad, sino, sobre todo, en su filosofía de funcionamiento. La programación orientada de objetos —en la que cada elemento del paquete de software contiene los detalles de su funcionamiento y de la influencia que ejerce sobre los otros objetos o elementos— lo hace apto para la producción de programas que no necesiten demasiada experiencia o aptitud en informática por parte de los usuarios. Es un hecho que este método de programación será empleado en prácticamente todos los ordenadores de la quinta generación que están en vías de desarrollo. La programación orientada de objetos es considerada de manera unánime como el adelanto más importante en el campo de la ciencia del software, desde que los lenguajes de alto nivel fueron introducidos a fines de los años cincuenta.

La mayor parte de los ordenadores personales tienen una capacidad de memoria suficiente y potencia de procesamiento para cubrir las necesidades del usuario. Toda ampliación de esta capacidad y potencia tiene como finalidad permitir una mayor comodidad en el manejo del ordenador por parte del usuario. Uno de los rasgos más notables del PCS es su habilidad para alcanzar un alto grado de simplicidad funcional con sólo 48 Kbytes.

Si bien la programación orientada de objetos se aplica de forma directa a juegos y otros programas gráficos, ésta requiere algo más de inventiva al programar si se desea aplicarla en el campo del software de gestión. Aunque no empleen los gráficos como medio principal de comunicación, los paquetes de hojas electrónicas son, en cierta manera, objetos en los que cada campo o celda puede contener unos datos y las relaciones que los definen.

Otro ejemplo lo constituye el sistema Lisa, de la firma Apple, que utiliza un "ratón" para desplazar un cursor por la pantalla a fin de seleccionar el programa (representado por un símbolo gráfico) que se desea hacer funcionar. La palabra procesador, por ejemplo, está representada por una hoja de papel en blanco, y el programa de trazado de gráficos, por una hoja de papel cuadriculado.

Quizá la más fascinante de todas sus funciones es el método que usa Lisa para trasladar datos de un programa a otro. Uno de sus *iconos* (representaciones pictóricas de funciones en la pantalla) es una tablilla con clips. Si se quiere tomar una pequeña sección de una hoja electrónica y reproducirla en forma de gráfica, sólo es necesario definir el recuadro sobre la hoja, trasladarlo a la tablilla con clips (que es un área de almacenamiento temporal) y a continuación transportarlo al programa trazador de gráficos.

Cuando se habló de juegos recreativos (véase p. 221), se puso de manifiesto la gran cantidad de juegos de diferente tipo que existían. El PCS podría muy bien entrar a formar parte de un nuevo grupo.

Llegados a este punto, es lógico suponer que el próximo paso que dará la industria de los juegos para ordenadores personales será la producción de este tipo de máquinas de que hablamos, aplicadas a laberintos, invasores del espacio, etc.; a partir de entonces, muchos escritores de programas para juegos podrán considerarse innecesarios.

Sistema de control del Cruise

Los controvertidos misiles Cruise, de fabricación norteamericana, poseen una interesante tecnología, como la memoria de burbuja, que pronto incorporarán los ordenadores personales

Cuando Neil Armstrong dio su primer paso sobre la Luna, ello fue posible, en gran medida, gracias a los sistemas de guía computerizados. Sin duda, la cohetería interplanetaria se basa en una ingeniería de alta precisión, pero sin el hardware y software de los ordenadores, no hubiera sido posible realizar los cálculos de posición con la suficiente rapidez y precisión como para permitir a un objeto entrar en contacto con otro a tan gran distancia, incluso con un cuerpo de las dimensiones de la Luna.

Cuando se tienen en cuenta las necesidades militares actuales, que exigen dar en un blanco con un margen de error limitado a 20 o 30 m después de sobrevolar un continente entero, la envergadura de la potencia de procesamiento de datos necesaria para realizar los cálculos resulta enorme.

La experiencia militar anterior mostró que el problema fundamental de los misiles era que, una vez lanzados, no era posible corregir su trayectoria. El primer avance de importancia llegó con el desarrollo de sistemas de guía sencillos que permitían determinar dónde estaba el cohete en relación con un punto de la superficie de la Tierra (el lugar de lanzamiento), deduciendo la distancia recorrida y en qué dirección. Pero incluso un sistema de este tipo adolecería de un margen de error importante.

Otro método de mayor precisión emplea satélites situados en una órbita geoestacionaria como puntos de referencia. El principal inconveniente que presenta este sistema es que la trayectoria de vuelo del misil —y probablemente su blanco— pueden ser determinados por el enemigo al poco tiempo de su lanzamiento, debido a los modernos sistemas de radar. Para combatir esta vulnerabilidad, el requisito ideal era un misil que volara a muy baja cota y dotado de un pequeño radar de sección transversal que pudiera decidir por sí mismo la trayectoria a seguir hasta alcanzar su blanco. Y por ello nació el misil Cruise, que actualiza constantemente su posición, analizando el relieve del terreno sobre el que vuela. Esto se realiza haciendo coincidir una sucesión de lecturas de las alturas sobre el suelo, mediante un altímetro-radar muy preciso, con un mapa de curvas de nivel del terreno almacenado en una memoria de burbuja incorporada.

Este sistema, desarrollado por McDonnell Douglas, se conoce como *TERCOM* (*TERrain COntour Matching*: comparación de la curva de nivel del terreno), o DPW-23. Cada misil almacena en su memoria de burbuja unos 25 "perfiles de trayectoria", que compara con el terreno cuando vuela sobre él. Sin embargo, existen algunos inconvenientes. Por ejemplo, el sistema no se puede utilizar sobre el agua. Tampoco posee una precisión fiable sobre la arena del desierto, en constante movimiento, ni en pleno invierno en los países del norte de Europa, debido a las notables alteraciones del terreno por las prolongadas nevadas.

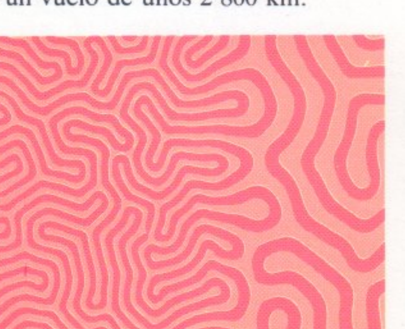
El Cruise no utiliza este sistema de guía desde el momento de su lanzamiento. Permanece inactivo mientras vuela alto en el espacio aéreo no peligroso. Cuando puede sufrir un ataque desde el aire o tierra, entra en picado hasta situarse a 15 m sobre el suelo, para su vuelo sobre territorio enemigo. Aunque en este punto pueda haberse desviado hasta 1 km de su trayectoria correcta, podrá volver a ella con precisión gracias a uno de sus 25 mapas.

Cuando el misil se encuentra cerca del blanco, conecta una unidad terminal de comparación, que contiene —también en una memoria de burbuja— un grabado digital detallado del área del objetivo. Las pruebas han demostrado que este sistema es capaz de tener una precisión notable, con un margen de error de apenas 18 m de distancia respecto al blanco, tras un vuelo de unos 2 800 km.



Cortesía Aerospace Publishing Ltd.

Misil autodirigido
El misil Cruise tierra-tierra de la General Dynamics "Tomahawk" tiene una longitud de 6,40 m y pesa menos de una tonelada y cuarto (1 200 kg). Se dispara desde un tubo montado sobre una base móvil. Su trayectoria empieza como la de un cohete convencional, pero al poco tiempo despliega unas pequeñas alas y se sitúa a pocos metros del suelo. Es propulsado por un compacto y diminuto turborreactor

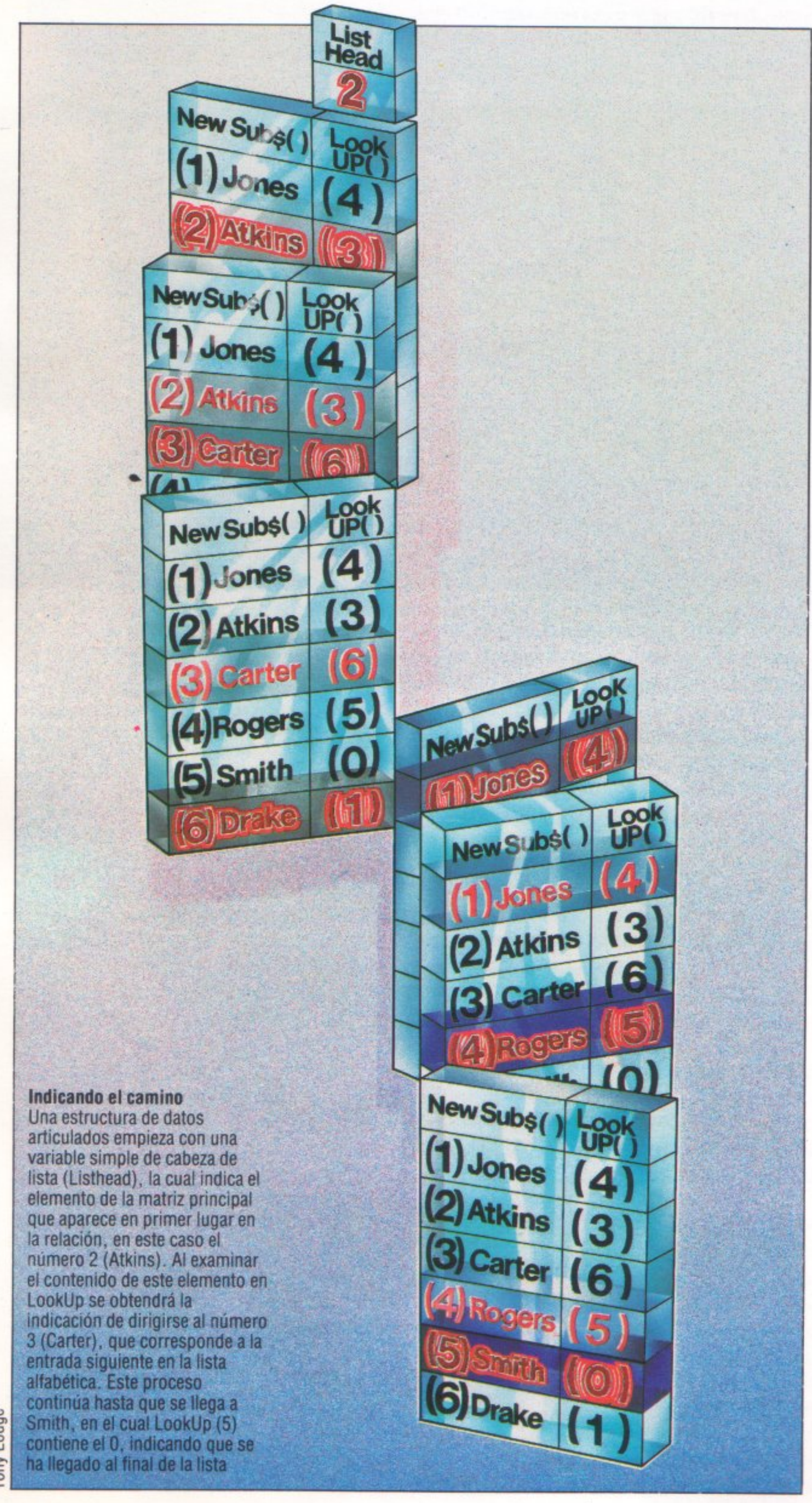


Cortesía New Scientist

Memorias de burbuja
En este tipo de memoria se obtiene un "1" cuando se crea una "burbuja" de fuerza magnética, mientras que su ausencia representa el "0", en un diminuto chip de granate. Las ventajas que presenta son su densidad (un millón de bits, o 128 Kbytes por chip) y la no pérdida de contenido cuando se desconecta la fuente de energía. Pero las memorias de burbuja reaccionan más lentamente que las RAM convencionales

Cota de malla

Clasificar es una manera de estructurar grandes cantidades de datos, por ejemplo, nombres y direcciones. La lista articulada o cadena es una forma alternativa que presenta claras ventajas



Indicando el camino
Una estructura de datos articulados empieza con una variable simple de cabeza de lista (Listhead), la cual indica el elemento de la matriz principal que aparece en primer lugar en la relación, en este caso el número 2 (Atkins). Al examinar el contenido de este elemento en LookUp se obtendrá la indicación de dirigirse al número 3 (Carter), que corresponde a la entrada siguiente en la lista alfabética. Este proceso continúa hasta que se llega a Smith, en el cual LookUp (5) contiene el 0, indicando que se ha llegado al final de la lista

Tony Lodge

En la memoria de un ordenador sólo hay datos, byte tras byte de información, almacenados como diferencias de potencial. Estos bytes adquieren su significado por la estructura de los datos que impone el procesador central. Esas diversas formas deciden si un byte determinado debe ser interpretado como componente de una instrucción, o como dígito de un número, o como un código de un carácter.

El usuario del ordenador parte del hecho de que algunos tipos de estructuras están prácticamente incorporados en el ordenador. Por lo general, los lenguajes de programación exigen que los datos sean dispuestos en un número limitado de formas. El lenguaje BASIC impone la idea de tipos numéricos y series de datos, y suministra variables y formas matriciales para manipular estos datos. Otros lenguajes soportan esas estructuras y otras adicionales. La fuerza y variedad de sus tipos de datos son los componentes principales para la determinación de la potencia de un lenguaje. Las estructuras de datos en BASIC —variables y matrices— será lo único que se necesite para simular otras formas de considerar el significado de los datos.

La matriz de clasificación es una estructura de datos práctica, y fácilmente realizada en BASIC. Sin embargo, tiene sus limitaciones, en particular cuando los datos de referencia cambian con frecuencia y/o imprevisiblemente. Supongamos que British Telecom posee un archivo de sus nuevos suscriptores para su inclusión en la próxima edición del listín telefónico. Hasta ese momento, los nombres y direcciones deben mantenerse en orden alfabético, para que puedan ser manejados con facilidad. Pero el archivo crece constantemente, y estos nuevos datos llegan de una manera imposible de predecir; y así, un lunes cualquiera, el archivo New Sub$ ( ) ("nuevos suscriptores") podría tener la siguiente forma:

NewSub$ ( )	Index ( )
(1) Jones	(2)
(2) Atkins	(3)
(3) Carter	(6)
(4) Rogers	(1)
(5) Smith	(4)
(6) Drake	(5)

La matriz Index ( ) muestra el orden en el que leer NewSub$ ( ), de manera que las entradas se encuentren en orden alfabético. Así, el primer suscriptor sería NewSub$ (2), Atkins. El segundo NewSub$ (3), Carter. En este ejemplo sólo se indican los nombres, pero, de hecho, una entrada en el listín comprendería el nombre y dirección —por lo general, unos 60 caracteres—. Desplazar bloques de 60 caracteres a través de la memoria es lento (puesto que su orde-

nación requiere numerosos movimientos de datos); por tanto, es más práctico dejar NewSub$ ( ) sin clasificar, y crear, en su lugar, un Index ( ). Ahora supongamos que se debe añadir un nuevo nombre, Bull, al archivo; la matriz quedará de esta manera:

NewSub$ ( )	Index ( )
(1) Jones	(2)
(2) Atkins	(7)
(3) Carter	(3)
(4) Rogers	(6)
(5) Smith	(1)
(6) Drake	(4)
(7) Bull	(5)

Nótese que el contenido de Index ( ) posterior al nuevo elemento no ha cambiado, y los elementos anteriores están en el mismo orden que antes, pero todos han sido desplazados un lugar. Por tanto, la inserción de un nuevo dato requiere hallar la posición del nuevo elemento, incrementar en una unidad cada elemento comprendido entre aquél y el final del índice, y escribir la nueva entrada. Esto es preferible a hacer lo mismo con los datos reales, NewSub$, pero es aún relativamente lento si el índice es largo.

Supóngase ahora que se estructuran los datos de forma distinta. Se deja sin clasificar NewSub$ ( ), debido a que su manipulación es lenta y cara, y se establece una matriz paralela llamada LookUp ( ), cuyo contenido son simples números que hacen referencia a las posiciones en NewSub$ ( ).

ListHead (2)

NewSub$ ( )	LookUp ( )	Index ( )
(1) Jones	(4)	(2)
(2) Atkins	(3)	(3)
(3) Carter	(6)	(6)
(4) Rogers	(5)	(1)
(5) Smith	(0)	(4)
(6) Drake	(1)	(5)

La primera diferencia es que se necesita una variable simple, denominada ListHead: hace referencia a NewSub$ (2), que es alfabéticamente el primer elemento de NewSub$ ( ). La siguiente diferencia es que se ha utilizado el número (0) en LookUp (5): esto indica que NewSub$ (5) es alfabéticamente el último elemento de la matriz.

Otra diferencia es el contenido de Index ( ) y LookUp ( ). La interpretación de Index ( ) debe ser: "el primer elemento está en NewSub$ (2), el segundo en NewSub$ (3), el tercero en NewSub$ (6)"... etc. Mientras que la de ListHead ( ) sería: "el primer elemento está en NewSub$ (2)". Luego LookUp (2) dice que el próximo elemento está en NewSub$ (3); LookUp (3) indica que el siguiente se encuentra en NewSub$ (6); y así sucesivamente. LookUp (5) informa que NewSub$ (5) es el último elemento.

Index ( ) da una posición absoluta para los elementos del archivo, mientras que LookUp ( ) proporciona sólo posiciones relativas: un componente de LookUp ( ) únicamente dice dónde encontrar el próximo elemento, y no proporciona ninguna información sobre la posición absoluta. El número en Index (4) indica el cuarto componente del archivo ordenado alfabéticamente, mientras que el número en LookUp (4) hace referencia sólo al elemento que aparece tras NewSub$ (4) en el archivo ordenado. LookUp ( ) lleva a cabo la estructura de datos llamada *lista escalonada*. Leer una lista de este tipo es como partir en una expedición "en busca del tesoro": al principio se conoce sólo el primer punto adonde ir. Cuando se ha alcanzado éste, se encuentra una pista que indica el próximo destino, y así sucesivamente. Leer una matriz de clasificación es como participar en un rally de coches: al principio se reciben indicaciones de todos los puntos a los que hay que ir y el orden que hay que seguir.

La gran ventaja de la estructura de lista es su flexibilidad. Véase la lista tras insertar el nuevo elemento, Bull:

ListHead (2)

NewSub$ ( )	Index ( )
(1) Jones	(4)
(2) Atkins	(7)
(3) Carter	(6)
(4) Rogers	(5)
(5) Smith	(0)
(6) Drake	(1)
(7) Bull	(3)

La matriz LookUp ( ) sólo cambia en dos lugares:

a) LookUp (2), que antes hacía referencia a NewSub$ (3) como poseedor del próximo elemento alfabético después de NewSub$ (2), y ahora indica hacia NewSub$ (7), puesto que éste es ahora el siguiente elemento por orden alfabético después de NewSub$ (2)
b) LookUp (7), que no se había utilizado, hace referencia a NewSub$ (3) como próximo componente tras NewSub$ (7) en la ordenación alfabética.

Esto ilustra sobre el proceso general de inserción en una lista escalonada: encontrar el elemento de la lista que debe ir antes del elemento nuevo; luego, hacer que aquél haga referencia a este último, y que el nuevo elemento indique el componente que ha sido desplazado. Estas operaciones sencillas será lo único que se requiera para la inserción en una lista escalonada, y sólo la primera de ellas es afectada por el tamaño de la lista. Insertar un elemento en la lista es como introducir un nuevo eslabón en una cadena: decidir dónde hay que poner el eslabón, abrir la cadena, unir el eslabón precedente al nuevo, y éste al posterior. A veces, las listas escalonadas se conocen como *listas encadenadas*. Los números de LookUp ( ) —los eslabones— se denominan también *indicadores*.

Una sorprendente cualidad de las listas es su acusada serialidad; es imposible encontrar un elemento si no se parte desde el principio y se inspecciona cada elemento hasta hallar el adecuado. La lista, en este caso, se realiza mediante el uso de matrices, las cuales están proyectadas como estructuras de acceso directo, pero, de hecho, la lista las ha convertido en archivos secuenciales. En otros lenguajes, como, por ejemplo, el LISP y el PASCAL, esta característica está incorporada en el ordenador. Las listas son estructuras prácticas para el manejo de datos dinámicos (datos que cambian con regularidad), y pueden constituirse en eficaces herramientas cuando se trata con lenguajes naturales (como reconocimiento de la voz) o artificiales (compilación de programas), donde los mismos datos forman de manera natural una lista de elementos.

Presentamos "Sonido...

"Sonido y luz" es una nueva sección que le enseñará a obtener el máximo partido del sonido y de los gráficos de su ordenador personal

A medida que los ordenadores personales se han ido perfeccionando durante los últimos años, sus configuraciones se han vuelto mucho más versátiles y eficaces. Los juegos han tenido una influencia decisiva en la popularidad obtenida por cada nuevo ordenador, y se ha dedicado mucho tiempo y esfuerzos a desarrollar sofisticados gráficos en color. Aunque a primera vista no parezcan tan importantes, las características de sonido y elaboración de música han sufrido un desarrollo similar. Si se preguntara a algún escritor de programas de primera línea cuál es la importancia de las rutinas de sonido en sus programas, probablemente la situaría en un tercer lugar, tras el concepto del juego y de los gráficos. Un uso inteligente de los efectos sonoros y de la música contribuye en gran manera al entretenimiento y al interés de los juegos recreativos.

Además de las aplicaciones para juegos, cabe la posibilidad de ampliar los conocimientos musicales mediante la utilización de las características de sonido que poseen los ordenadores personales. En muchos casos se dispone de órdenes especiales para música en lenguaje BASIC, que permiten escribir programas breves para interpretar melodías bastante complejas que hasta pueden incluir acordes. Algunos ordenadores poseen también formas de cambiar la naturaleza del sonido para hacerlo más agradable al oído o convertirlo en algo parecido al de los instrumentos musicales convencionales. En todos los casos, el teclado del ordenador puede ser configurado mediante un programa adecuado, de forma que actúe como el teclado de un piano, permitiendo tocar música a "tiempo real".

Incluso en el caso de que se tengan pocos conocimientos de programación, es posible escribir programas cortos y sencillos que produzcan sonidos musicales relativamente complejos. Si se desea obtener el máximo partido de las características de sonido, muchas casas de software producen programas extensos de música que permiten escribir y tocar melodías inmediatamente. Para cualquier tipo de utilización que se haga, es conveniente que el usuario comprenda cómo el ordenador genera la producción de sonido, le da forma y la controla.

...y luz"

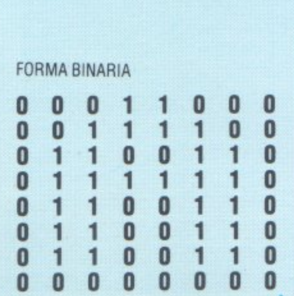
Alta y baja resolución

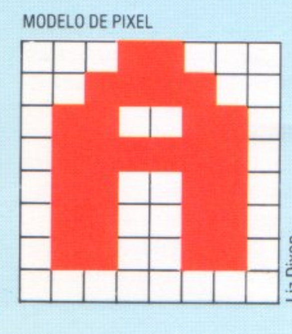
Los gráficos de los microordenadores pueden dividirse en dos categorías: de alta y baja resolución. La mejor forma de describir la diferencia entre ambas es considerar cómo se forma un carácter (una letra, número o figura).

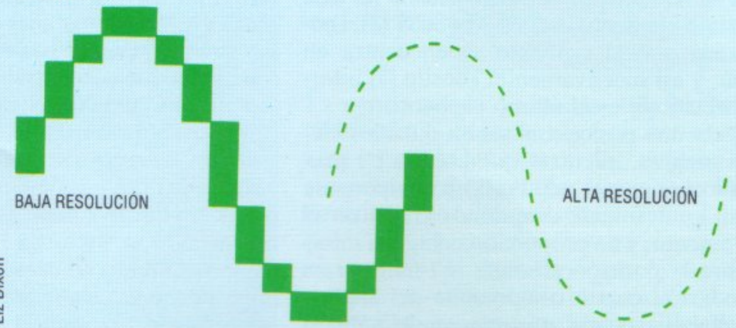
Si se mira desde cerca un carácter normal impreso en una pantalla de televisión, se puede ver que está compuesto por un grupo de pequeños cuadros. Estos cuadros reciben el nombre de *pixels* (contracción de *picture cell*: unidad pictórica), y cada carácter o figura que aparece en la pantalla es una ordenación de aquéllos. En la mayoría de los ordenadores personales, los caracteres se forman a partir de un cuadrado de 64 pixels, agrupados en ocho filas y ocho columnas. La letra "A" puede lograrse de un modelo de pixel similar a éste:

Cada pixel iluminado en el cuadriculado puede representarse en la memoria por un "1", y cada pixel oscuro por un "0". Ocho bits crean un byte, por tanto cada fila del cuadriculado del carácter puede almacenarse en una sola localización de la memoria. Así, para obtener un carácter simple será preciso ocupar ocho localizaciones de memoria.

A veces, la visualización de los gráficos se construye con bloques del tamaño completo, medio o cuarto del cuadriculado de un carácter. Los gráficos proyectados empleando estos bloques sencillos y grandes se dice que son de baja resolución. En muchos ordenadores domésticos es posible diseñar gráficos compuestos por pixels simples. Éstos corresponden a visualizaciones de alta resolución. Una buena manera de demostrar la diferencia entre los dos tipos es comparar las siguientes curvas senoidales, obtenidas mediante ámbos tipos de resolución.







Osciladores

Los osciladores son circuitos electrónicos que producen señales repetitivas. Cuando estas señales son amplificadas y enviadas a un altavoz, producen sonidos de un tono dado. El número de osciladores que posee un ordenador personal puede variar entre uno y cuatro: cuantos más osciladores haya, más notas podrán tocarse al mismo tiempo.

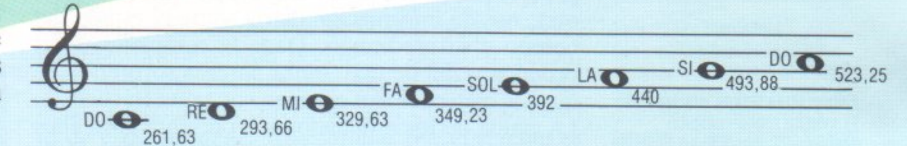
El sonido creado se define por tres características: frecuencia, envoltura (que incluye el volumen) y forma de la onda. La primera de ellas la estudiaremos en esta ocasión, y las otras dos las trataremos en el próximo capítulo.

Frecuencia

Ésta es la característica más importante que es necesario controlar, pues determina el tono del sonido. La frecuencia se define como el número de veces que se repite una señal en un segundo y su unidad de medición es el hertz (Hz, ciclos por segundo). Los sonidos que puede percibir el oído humano están comprendidos entre 20 y 20 000 Hz. Aunque no podamos oír frecuencias por debajo de los 20 Hz, éstas pueden utilizarse para modificar las características de un sonido audible. Esta técnica recibe el nombre de *modulación*, y entre los ordenadores personales, en la actualidad, sólo se puede aplicar en el Commodore 64.

Sin embargo, no es preciso profundizar más en la explicación de la frecuencia. Lo que es necesario conocer es cómo tocar las notas musicales. La facilidad con que pueda hacerse esto varía mucho de una máquina a otra. Algunas poseen órdenes en BASIC que determinan por sí mismas la frecuencia, y, por tanto, lo único que hay que especificar es el número de tono o incluso nada más que la nota musical: *la, la #, si,* etc. En otras, sin embargo, es mucho más difícil, pues sólo suministran una tabla en el manual del usuario, en la cual debe buscarse la frecuencia necesaria para cada nota y, mediante la orden POKE, introducir su valor en una posición de la memoria. La tabla proporciona los valores de conversión exactos para la escala en *do mayor*. Esta tabla también es útil si se desea programar música en código máquina, pues el lenguaje BASIC no sirve para ayudar a calcular las frecuencias.

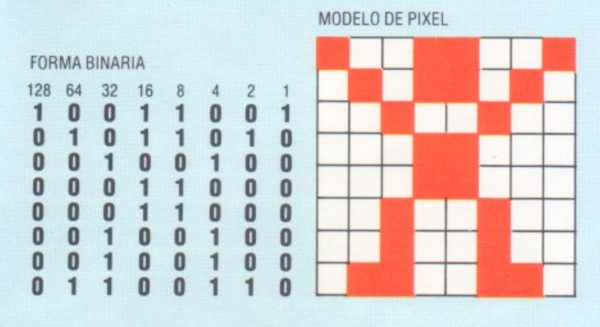
Frecuencias de las notas musicales
Se puede determinar la frecuencia de cada nota de la escala musical multiplicando la de la nota situada un semitono más abajo por 1,0594631. Esto puede parecer un poco complicado, pero si se efectúa la multiplicación 12 veces, se obtendrá una frecuencia doble de la primera. En una octava hay 12 semitonos. Por tanto, al doblar la frecuencia se logra un sonido de una octava superior. La tabla adjunta proporciona las conversiones exactas de las notas musicales (para la escala en *do mayor*) a sus frecuencias



Caracteres definidos por el usuario

Para crear en la pantalla visualizaciones atractivas y poco comunes, con frecuencia es útil tener disponibles caracteres que sean diferentes a los del conjunto alfanumérico normal. El Vic-20 y el Commodore 64 tienen un juego especial de formas gráficas que puede emplearse directamente con el teclado, pero incluso éstas no cubren todas las posibilidades. En la mayor parte de los ordenadores personales es posible crear caracteres nuevos. Esto, por lo general, se logra volviendo a definir las formas binarias de las ocho localizaciones de memoria en que está almacenado un carácter.

Durante el proceso, el conjunto antiguo de formas binarias a menudo se pierde, y el "definido por el usuario" adquiere algunas de las propiedades del sustituido en la memoria. Así el nuevo puede usarse con la orden PRINT, simplemente apretando la tecla del carácter que ha sido reemplazado. A continuación se muestra un ejemplo, junto con sus códigos binarios asociados:



128	64	32	16	8	4	2	1
1	0	0	1	1	0	0	1
0	1	0	1	1	0	1	0
0	0	1	0	0	1	0	0
0	0	0	1	1	0	0	0
0	0	0	1	1	0	0	0
0	0	1	0	0	1	0	0
0	0	1	0	0	1	0	0
0	1	1	0	0	1	1	0

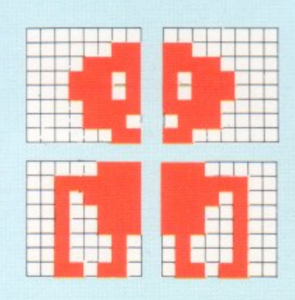
La facilidad con que se pueden establecer varía mucho según el tipo de ordenador que se utilice. Por ejemplo, en el Sinclair Spectrum, con la orden USR, lo único que se necesita es introducir las formas binarias apropiadas; sin embargo, en el Commodore 64, el usuario, en primer lugar, tiene que desplazar todo el conjunto de caracteres desde ROM a RAM antes de (mediante la orden POKE) introducir en la memoria los ocho equivalentes decimales de las formas binarias que crean la figura. No obstante, es posible adquirir, a través de distribuidores independientes, cierto número de programas de diseño de caracteres que pueden facilitarle de manera notable la labor al usuario del ordenador Commodore 64.

Para crear figuras más grandes, es posible agrupar dos o más caracteres. La figura del "extraterrestre" que podemos ver en la esquina inferior derecha está formada con cuatro caracteres definidos por el usuario. El programa en funcionamiento en el Commodore 64 imprime el grupo en la pantalla en tres colores diferentes. Los caracteres se crearon mediante una rutina corta para desplazar el conjunto normal de ROM a RAM y sustituir las formas gráficas ▞, ◨, ◪, y ‛ ’ leyendo números decimales de sentencias DATA y utilizando órdenes POKE para colocarlos en las posiciones apropiadas. Este tema lo trataremos con mayor profundidad en próximos capítulos.

Incluso cuando se dispone de "sprites" (véase p. 152), frecuentemente existe un límite en el número que se puede visualizar en la pantalla en cada momento. Por eso los gráficos definidos por el usuario cobran toda su importancia cuando es necesario representar muchas figuras similares a un mismo tiempo.


Ian McKinnell

Extraterrestres
Estas raras criaturas se crearon a partir de cuatro caracteres, cada uno de ellos definido por el programador. Este sistema puede usarse en máquinas que no posean "sprites"



Pronósticos más precisos

El uso de ordenadores de alta velocidad para procesar imágenes de satélites y analizar las estructuras de datos ha permitido que las predicciones meteorológicas sean mucho más precisas

Los resultados de muchas de las tareas de proceso de datos más complejas están presentes en nuestra vida cotidiana, con frecuencia sin que seamos conscientes de ello. Una de las aplicaciones más avanzadas del ordenador, que requiere una capacidad de procesamiento de datos mayor que casi cualquier otra, nos da información diaria sobre el estado del tiempo y su predicción. Dada la complejidad del pronóstico del tiempo, quizá parezca sorprendente que los meteorólogos acierten con tanta frecuencia. La ayuda de los ordenadores representa para ellos una inmensa ventaja en el manejo de tan vasta gama de posibilidades.

Los factores climatológicos que afectan al tiempo en las regiones occidentales europeas son muy complejos. En primer lugar, están condicionados por la proximidad al polo Norte y al océano Atlántico. Al estar situadas al este del Atlántico, las regiones occidentales europeas son más proclives a los fenómenos climatológicos creados en dicho océano, debido al efecto Coriolis, fenómeno que se debe al giro de la Tierra de oeste a este. Esto se comprenderá mejor si recordamos que en el ecuador un objeto situado en la superficie de la Tierra viaja a más de 1 600 km/h; y este potente movimiento de rotación, combinado con las corrientes de aire normales del polo al ecuador, crea los vientos predominantes del oeste en el hemisferio Norte. Es este constante ataque de aire húmedo —que aumenta o disminuye según las variaciones locales de temperatura— el que determina las condiciones climatológicas en el oeste de Europa.

Para sus predicciones, los servicios de meteorología cuentan con estaciones de recogida de datos situadas en lugares estratégicos del Atlántico —bar-



Fotografías desde el espacio
El satélite meteorológico Meteosat 2, lanzado en junio de 1981, está situado en una órbita geoestacionaria (es decir, no se mueve con relación a la Tierra) a unos 35 880 km sobre el ecuador, en el meridiano cero. Reúne información de un gran número de estaciones terrestres

cos meteorológicos, boyas, globos y aviones de reconocimiento—, que suministran datos sobre las condiciones más inmediatas. Y con esta información se predice qué sucederá cuando los fenómenos climatológicos originados en el océano lleguen a tierra firme, de acuerdo con el comportamiento de fenómenos anteriores.

Antes de marzo de 1979, cuando el satélite meteorológico Meteosat I fue puesto en órbita, el único método de predicción disponible por los hombres del tiempo era trasladar los informes de las estaciones climatológicas a un mapa para formar una gráfica isobárica. Las isobaras son líneas imaginarias que unen puntos con igual presión barométrica, de la misma forma que las líneas de nivel de un mapa unen puntos de la misma altura. Basándose en las isobaras, es posible determinar la velocidad y dirección de frentes fríos o cálidos —y, en asociación con ellos, de borrascas y anticiclones— y de esta forma predecir el estado del tiempo.

Si bien los mapas isobáricos son sin duda los más utilizados, no son, sin embargo, los únicos que se trazan en los diversos institutos meteorológicos. A partir de la base de datos climáticos que poseen sus sistemas de ordenadores, éstos levantan mapas que muestran temperaturas medias, precipitaciones, horas de sol por día, etc.

Procesamiento de números

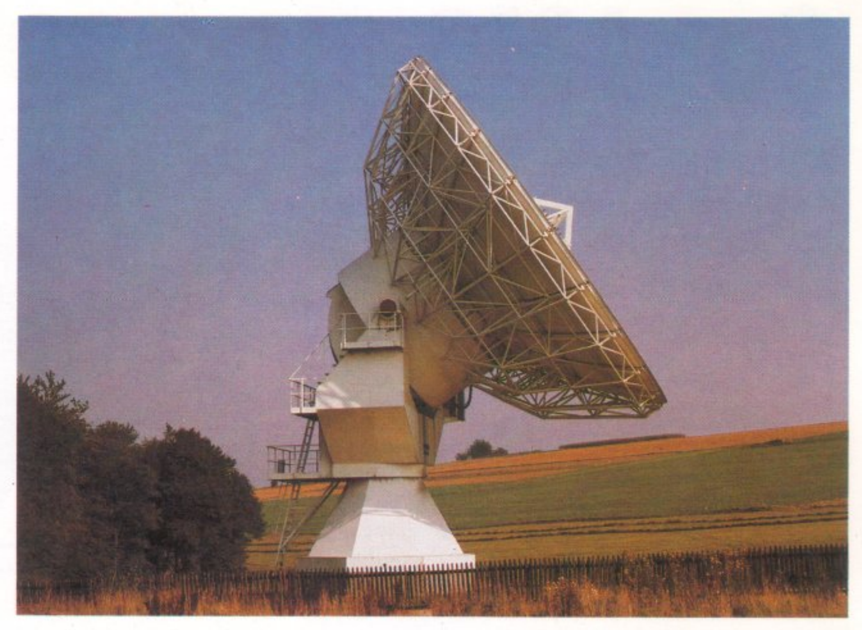
Uno de los principales usos de los grandes ordenadores en la investigación científica consiste en el procesamiento de información exclusivamente numérica en forma de ecuaciones muy extensas y complejas. Las aplicaciones puramente científicas, como por ejemplo de física nuclear, o de ciencia aplicada, como las de meteorología, exigen requisitos similares. Si bien existe la posibilidad de llevar a cabo cálculos de este grado de complejidad con un microordenador, el tiempo necesario para ello sería excesivo, como consecuencia no sólo del número de términos de la ecuación, sino por la desmesurada magnitud de los números a tratar. Para poder realizar esta operación en un tiempo razonable, es necesario disponer de ordenadores muy rápidos y con gran capacidad de memoria

Los servicios meteorológicos siguen aún este procedimiento para trazar sus precisos mapas del estado del tiempo, pero hoy en día emplean también las imágenes recibidas desde el Meteosat. Éstas son señales analógicas que son digitalizadas por el ordenador para su procesamiento y visualización en forma de mapas coloreados artificialmente. Las imágenes crean un vívido panorama de la composición del tiempo en ese momento. Estos mapas se rehacen cada cuatro minutos aproximadamente, y, por lo tanto, el meteorólogo puede observar las variaciones climáticas en tiempo real.

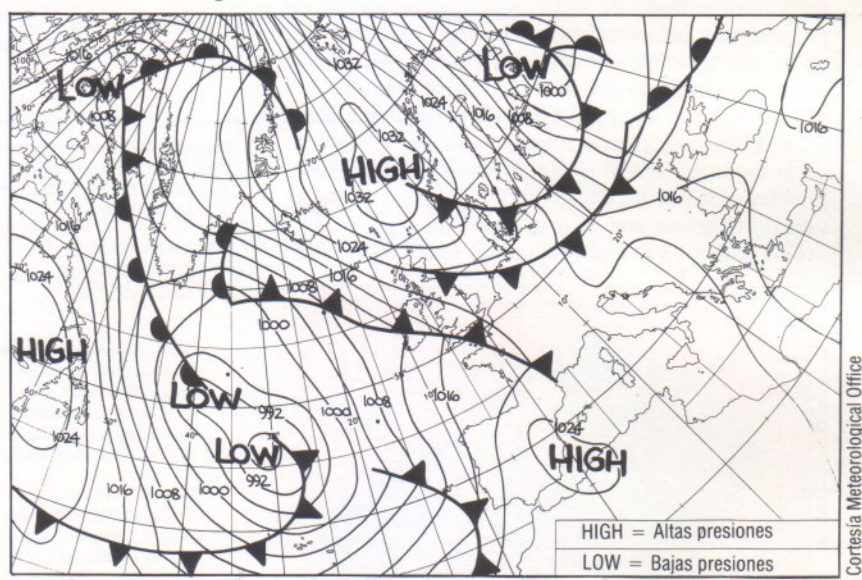
El Meteosat 2, que reemplazó al satélite anterior en junio de 1981, está situado en una órbita geoestacionaria a unos 35 880 km sobre el ecuador, en el meridiano cero. Reúne información que le proporcionan un gran número de estaciones terrestres esparcidas por toda la superficie del globo, que es transmitida a quienquiera que desee suscribirse al sistema.

Teóricamente sería posible analizar e interpretar esta información (aunque no en tiempo real) mediante un ordenador personal, escribiendo en un disco los datos que se reciben del satélite. Sin embargo, la señal es de tipo analógico, y, por consiguiente, la conversión podría ser difícil. También sería necesario instalar una antena parabólica alineada exactamente con el satélite. El procesamiento de estas imágenes transmitidas por el satélite es sólo una función muy pequeña del sistema de ordenadores del Servicio Meteorológico Nacional. Junto con otras organizaciones de este mismo carácter localizadas en otras partes del mundo, mantiene un modelo del sistema del tiempo global y extrae de éste una gran cantidad de datos estadísticos. Éstos forman la base de datos de la información histórica a partir de la cual se pueden pronosticar tendencias meteorológicas y establecer las características del clima local y global, que incluyen no sólo datos barométricos, sino también detalles sobre la velocidad y dirección del viento, precipitaciones y temperaturas al nivel del suelo y del mar.

El conjunto de estos datos tiene un valor determinante para el análisis histórico. De la misma manera, esta información es de importancia vital para muchas industrias, como asimismo para la economía y ecología de continentes enteros, pues únicamente por este medio pueden determinarse los cambios en el clima.



Estaciones terrestres
Las antenas parabólicas de recepción de señales emitidas desde satélites pueden ser muy diferentes en complejidad y tamaño. La que aparece aquí es capaz de recibir y transmitir señales, y no se limita a captar información desde los satélites geoestacionarios. Posee un sofisticado control por ordenador que le permite seguir la trayectoria de un satélite



Cortesía Meteorological Office

Cartas isobáricas
Los "mapas del tiempo" que vemos en televisión son en realidad cartas de la presión barométrica. Las líneas concéntricas unen puntos de igual presión atmosférica. Los vientos giran en dirección contraria a las agujas del reloj alrededor de un centro de "bajas" presiones, y en sentido inverso alrededor de un punto de "altas" presiones (al contrario en el hemisferio Sur), y la velocidad del viento es directamente proporcional a la distancia entre las isobaras

Sord M5

A pesar de que este ordenador, en su versión estándar, posee sólo cuatro Kbytes de memoria, sus magníficas características gráficas permiten escribir valiosos programas

La mayor parte de los ordenadores personales de la primera época fueron diseñados en California (Estados Unidos). Recientemente, también las máquinas proyectadas en Gran Bretaña han empezado a conquistar buena parte del mercado mundial. Sin embargo, es previsible que dentro de poco tiempo los japoneses dominen la escena, tal como ha sucedido en otros sectores del mercado de la electrónica. Ciertamente, el Sord M5 no es el primer microordenador japonés, pero sí ha sido el primero en tener un éxito importante entre los ordenadores personales, a diferencia de lo sucedido en el mercado de los comerciales.

Es una máquina sólida y compacta, de tamaño similar al del Sinclair Spectrum, pero considerablemente más pesada y da la sensación de ser mucho más poderosa. En otros muchos aspectos tiene una capacidad similar, con una CPU Z80A, una sola tecla para entrada de BASIC, y almacenamiento de programas e información en cassette. No obstante, en su interior es mucho más sofisticado, tal como se pone de manifiesto por la conexión de la impresora Centronics que lleva incorporada la máquina. Pero las dos diferencias más importantes son el tamaño de la memoria RAM —que con sus 4 Kbytes (ampliables a 36) es mucho más pequeña— y la inclusión de chips de sonido y gráficos.

Los gráficos son elaborados por un TI 9918, 9928



Cartucho de ROM
Una de las características más sobresalientes del M5 es la posibilidad de cambiar de lenguaje, debido a que éste se encuentra almacenado en un cartucho de ROM. El M5 dispone de tres versiones de BASIC: BASIC-I (simple, para principiantes); BASIC-G (muy eficaz en gráficos), y BASIC-F (científico y matemático). Existe también un programa especial de uso general, llamado FALC, que consta de una combinación de hojas electrónicas, archivo y funciones gráficas, para desarrollar aplicaciones complejas para uso personal o de gestión

Conector impresora
En este punto se conecta una interface en paralelo de Centronics, que permite acoplar directamente al M5 una amplia gama de impresoras

Conector RF
Aquí se conecta la salida compatible de TV

Modulador
La salida del VDP es convertida en una señal estándar de TV

Conector video
La señal de video, compuesta y sin modular, puede utilizarse en un monitor

Conector audio
Desde este enchufe puede alimentarse un amplificador con la señal de audio

VDP
El Video Display Processor TMS 9929 de la Texas es el responsable del control de la pantalla y puede manipular hasta 32 "sprites" independientes

Conectores mandos de juegos
En este punto se conectan los dos mandos para juegos

RAM video
Todos los datos necesarios para controlar la pantalla, incluso imágenes reales, son manipulados por este bloque de RAM de 16 Kbytes



Mandos de juegos
Son los equivalentes de las palancas de mando. Funcionan enviando una señal a cada uno de los cuatro puntos en diagonal. Estas señales interrumpen a la CPU, por lo que el tiempo de respuesta es muy breve

Conector cassette
Esta interface es un enchufe tipo DIN y tiene conexiones para controlar el motor de la grabadora

Conector potencia
Desde aquí se obtiene la fuente de alimentación, a través de un pequeño transformador

Chip de diseño propio
El M5 utiliza una pieza de sofisticado y exclusivo diseño para realizar sus funciones superiores a un precio razonable

ROM
Los únicos programas incorporados en la máquina son un conjunto de programas de control de bajo nivel, que son utilizados por el programa del usuario. Cuidan de los detalles del manejo de la pantalla, teclado y cassette

CPU
El procesador del Sord M5 es el conocido Z80A. Éste tiene una velocidad de reloj de 3,58 MHz

RAM
La memoria del usuario está contenida en estos dos chips de considerable tamaño, y es independiente de otras áreas de la RAM

Controlador automático
Gran parte de la limpieza de funcionamiento del M5 se logra mediante este avanzado controlador automático, que regula y coordina varias funciones de la máquina

o 9929 (según sea el país en que se vende el ordenador), que da una resolución de 192 × 256 puntos de hasta 16 colores diferentes. Hay cuatro modelos gráficos principales, tres de los cuales permiten tener hasta 32 "sprites" capaces de moverse independientemente, pudiendo ser de tamaño normal o ampliado. La máquina puede visualizar letras mayúsculas y minúsculas, signos de puntuación y caracteres numéricos. Posee símbolos de dibujo de líneas y bloques, así como una amplia gama de letras minúsculas acentuadas, y, como cada carácter puede redefinirse, las posibilidades que ofrece son muy amplias.

Otros ordenadores emplean los mismos chips de gráficos —en particular el TI99/4A (véase p. 189)— y es el uso de éstos lo que hace que el Sord M5 sea tan efectivo a pesar de su falta de memoria RAM. Puesto que la memoria de pantalla es totalmente independiente de la del programa, el único contenido de la RAM principal será el programa real, además, por supuesto, de los datos necesarios para las variables.

SORD M5

DIMENSIONES
185 × 70 × 55 mm

PESO
1 kg

CPU
Z80A

VELOCIDAD DEL RELOJ
3,58 MHz

MEMORIA
8 Kbytes de ROM
20 Kbytes de RAM, de los cuales 16 se utilizan para visualizaciones gráficas.
Con la adición de cartuchos, la ROM puede ampliarse a 16 Kbytes, y la RAM a 32

VISUALIZACION EN VIDEO
Hasta 16 colores, que pueden emplearse en diferentes "planos". Consta de gráficos "sprite" y cuatro tipos de pantalla diferentes: dos para gráficos, uno para texto y otro "multicolor"

INTERFACES
Cassette, impresora (Centronics), mandos de juegos, cartucho de ROM, audio

LENGUAJE SUMINISTRADO
El cartucho de lenguaje es el BASIC completo: BASIC-I

OTROS LENGUAJES DISPONIBLES
BASIC-G (gráficos), BASIC-F (BASIC de coma flotante), FALC (lenguaje para hojas electrónicas y base de datos)

VIENE CON
Adaptador de fuente de energía, cable cassette, cable televisión, dos palancas de mando con cables, un cartucho de lenguaje BASIC-I y una cassette con dos juegos

TECLADO
55 teclas: 8 teclas de cambio proporcionan todos los caracteres alfanuméricos, 28 funciones BASIC y 64 formas gráficas

DOCUMENTACION
Una guía del usuario en la que se describe la forma de conectar el ordenador, y cómo cargar y hacer funcionar los dos juegos; una página está dedicada a detectar errores simples. No existe descripción alguna sobre el lenguaje BASIC o del uso del cassette o de otras interfaces no relacionadas con los juegos suministrados

Un tema arduamente discutido en la actualidad en la industria de los ordenadores personales es el que se refiere a la propuesta de implantación de unas "normas MSX", desarrolladas por un grupo integrado por los fabricantes más impoortantes del Japón, incluida la Sord. La idea es que si los fabricantes se atienen a estas normas de estandarización en el diseño de los ordenadores personales (que afectan tanto al hardware como a la versión de BASIC a emplear), sería posible escribir software que funcionara en todas las máquinas, sin modificación alguna. Desde el punto de vista de los chips para gráficos, el Sord cumple estas normas.

Sin embargo, las MSX también especifican que el chip de sonido debe ser el AY-3-8910 de la General Instruments. Para producir los sonidos, el Sord M5 (al igual que el BBC Micro) utiliza un chip TI 76489, el cual tiene un mejor control sobre la gama de sonidos producidos que el chip GI, aunque se asemeja a éste al tener tres canales para tonos y uno para la creación de efectos especiales. Esto significa que el M5 no cumple todas las especificaciones que debería reunir una máquina MSX.

Con el cartucho de ROM pueden suministrarse tres versiones diferentes de BASIC, diversos complementos, juegos y varias aplicaciones, y como puede ampliarse hasta 16 Kbytes, es probable que aparezcan en el mercado programas útiles para esta máquina.

El teclado del Sord M5
El teclado de caucho es ligeramente mayor que el del Sinclair Spectrum, y su funcionamiento más suave le proporciona una mayor comodidad al usuario. Se pueden emplear un total de 55 teclas en varias formas, para obtener caracteres alfanuméricos, símbolos gráficos, o funciones en BASIC, mediante la tecla FUNC. Todas las teclas, mientras se mantengan apretadas, continuarán funcionando automáticamente, lo cual es muy práctico para procesar textos en pantalla



Chris Stevens

El control de paridad

La “paridad par” asegura que el número de bits 1 de un byte sea siempre par, lo que hace más fácil detectar errores de transmisión

Una de las principales ventajas de los ordenadores digitales, en comparación con los dispositivos analógicos, es que los errores e inexactitudes que tienen lugar en todos los circuitos eléctricos no se acumulan al pasar una señal a través de numerosos circuitos (véase p. 239). Sin embargo, si se transmiten datos a cierta distancia —ya sea mediante una interface serial y un par de cables, o a través de una línea telefónica—, el “ruido” eléctrico de fondo que exista en la línea puede ser suficiente para cambiar un bit sencillo de 0 a 1, o viceversa. Por lo general, el ordenador receptor no estaría en condiciones de detectar esta alteración y, en consecuencia, aceptaría el dato erróneo como correcto.

Veamos lo que sucede si un bit del código ASCII para la letra Q resulta alterado en la transmisión:

[ ] 1010001 (código ASCII transmitido para Q)
[ ] 1000001 (código ASCII recibido para A)

Un error como este en la transmisión de datos sería, como mínimo, un grave contratiempo y podría ser potencialmente catastrófico. Sin embargo, se recordará que los códigos ASCII se asignan únicamente a valores hasta 127, que requieren sólo siete bits (numerados de 0 a 6). El MSB (*Most significant Bit:* bit más significativo) —el bit siete— se emplea, en consecuencia, a menudo como un bit “de paridad”, con el fin de detectar si ha ocurrido un error.

Existen dos acuerdos para la utilización de bits de paridad: “paridad par” y “paridad impar”. Nosotros consideraremos la primera de ellas.

“Paridad par” significa que el bit de paridad (bit 7 en un código ASCII) está regulado de forma que el número total de bits 1 en el byte sea siempre un número par. Con la paridad par, las letras A y Q tendrán la siguiente composición:

[0] 1000001
(código ASCII para A con paridad par)
[1] 1010001
(código ASCII para Q con paridad par)

En el código ASCII para A hay dos bits 1; por lo tanto, el bit de paridad será 0 con el fin de que el total de los ocho bits sea par. En el código ASCII para Q existen tres bits 1; por consiguiente, el bit de paridad será 1. Esto hace que el número total de bits 1 sea cuatro, que es un número par.

Ahora veamos qué sucedería si el bit 4 de nuestro código ASCII para la letra Q fuera alterado como en el ejemplo anterior.

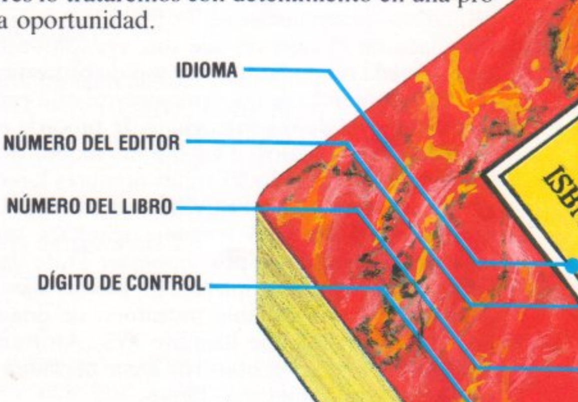
[1] 1000001 (código ASCII para Q alterado)

Al comprobar la paridad del byte (ya sea mediante software o por hardware especial), se ve que la Q correcta tiene un número par de “unos” (incluyendo el bit de paridad). La Q alterada, por el contrario, tendría el bit 4 accidentalmente cambiado de 1 a 0, pero el bit de paridad original —bit 7— continúa siendo 1. Al comprobar la paridad de este byte alterado, se encontraría un número impar de bits 1, y, por lo tanto, resultaría evidente que el byte estaba alterado y sería rechazado. Si se piensa sobre esto, se verá que incluso si el propio bit de paridad fuera el que se alterara en la transmisión, el hecho de que había ocurrido un error sería también detectado en el proceso de control de paridad, y, en consecuencia, el byte sería rechazado.

Si observa los códigos ASCII de cualquier ordenador, probablemente comprobará que el bit 7 (el MSB: el bit más significativo) se utiliza en realidad, pero no como bit de paridad. Se ha dispuesto así a fin de permitir que el ordenador tenga un juego de caracteres adicionales (en general, un juego de caracteres gráficos). Además, hay que considerar que los errores de transmisión de datos *en el interior* de un ordenador son muy poco frecuentes. Normalmente la paridad se emplea sólo cuando se transmiten datos a larga distancia, o cuando se graban datos en una superficie magnética (por ejemplo una cassette o un disco), que también es susceptible de incurrir en “errores de bits”.

El control de paridad es adecuado para indicar que un byte dado ha sido transmitido incorrectamente, pero no aporta información sobre cuál de los bits de un byte ha sido transmitido erróneamente, y, por consiguiente, el error no puede ser corregido por el ordenador que lo recibe. Incluso puede presentarse una circunstancia peor: si en un byte hay dos bits alterados, un byte incorrectamente transmitido puede ser tomado como correcto.

Sin embargo, en los casos en que el dispositivo receptor detecta un error, puede enviar un mensaje de error y el software, por su parte, puede dar las órdenes adecuadas para que el byte incorrecto se emita otra vez. Finalmente, debemos agregar que se han desarrollado sistemas muy sofisticados para la detección y corrección de errores; son capaces de indicar qué bit o bits han sido alterados, permitiendo, de esta manera, que sean corregidos automáticamente. El tema de los códigos de corrección de errores lo trataremos con detenimiento en una próxima oportunidad.



Sólo para controlar
El último dígito del International Standard Book Number (ISBN) tiene sólo funciones de control, equivalentes a las que desempeña el par de paridad en un ordenador. Si se multiplica el primer dígito (aquí 0) por 10, el segundo (5) por 9, y así sucesivamente, y se suma el resultado, se comprobará que el dígito de control se ha elegido de forma tal que el resultado sea divisible por 11

Clasificar y archivar

Como continuación de nuestro proyecto de programa para desarrollar una agenda computerizada, veremos cómo el archivo de datos deberá ser dividido en registros y campos

El capítulo anterior de nuestro curso de programación acababa tratando el tema de refinar los elementos de un ejercicio de programación mediante uno o más niveles de "seudolenguaje", hasta alcanzar un punto en donde los ejemplos pudieran ser codificados en BASIC. Empezaremos por revisar este ejercicio y dar algunas soluciones. La primera "enunciación de objetivos" para el ejercicio era:

INPUT
Un nombre (en cualquier formato)
OUTPUT
1. El nombre de pila
2. El apellido

En el primer nivel de refinamiento encontramos que podía ser dividido en seis etapas (más tarde vimos que la última podía omitirse). Éstas eran:

1. Leer el nombre completo (★LEER★)
2. Convertir en mayúsculas todas las letras (★CONVERT★)
3. Hallar el último espacio (★ESPACIO★)
4. Leer el apellido (★LEER APELL$★)
5. Leer el nombre de pila (★LEER NOMB$★)
6. Eliminar los caracteres no alfabéticos del nombre de pila

Todos estos temas se están tratando como subrutinas y el nombre que se ha asignado a cada una de ellas está descrito entre paréntesis. Desafortunadamente, la mayor parte de versiones de BASIC no pueden hacer referencia a las subrutinas por los nombres y será necesario, al escribir el programa final, insertar números de líneas después de los respectivos GOSUB. Durante la fase de desarrollo, sin embargo, resulta mucho más fácil referirse a ellas por el nombre. Posteriormente, estas denominaciones pueden incorporarse en sentencias REM. Este uso de subrutinas con nombres se indica colocando éstos entre asteriscos. En los lenguajes que pueden referirse a las subrutinas por el nombre (por ejemplo, el PASCAL), éstas son designadas, por lo general, como "procedimientos".

Incluso en el caso de que una versión de BASIC determinada no pudiera hacer uso de procedimientos, es recomendable que, mientras se está programando al nivel de seudolenguaje, se proceda como si pudiera emplearlos. Del mismo modo, alguna versión de BASIC no podrá tratar nombres largos de variables, tales como PROVINCIA o NOMBCALLE$, pero al nivel de seudolenguaje resultará mucho más fácil y claro darlo por supuesto. Trate de que los nombres sean descriptivos. Es mucho más claro denominar a una variable transitoria de una serie (string) VARTRAN$ que llamarle XV$. Afortunadamente, numerosas versiones de BASIC permiten usar nombres de variables más largos.

Ya hemos desarrollado la segunda de las etapas (convertir en mayúsculas todas las letras) mediante un segundo y tercer nivel de refinamiento, y se ha creado un programa corto en BASIC para realizar este cometido. Ahora efectuaremos lo mismo para las otras etapas:

```
2.º REFINAMIENTO
3. (Hallar último espacio)
BEGIN
LOOP mientras los caracteres no explorados
permanecen en NOMBRE$
  IF carácter = " "
    THEN anotar posición en una variable
    ELSE no hacer nada
  ENDIF
ENDLOOP
END
```

```
3.er REFINAMIENTO
3. (Hallar último espacio)
BEGIN
READ NOMBRE$
LOOP (mientras los caracteres no explorados
permanecen)
  FOR L = 1 TO longitud de NOMBRE$
  READ carácter de entre NOMBRE$
  IF carácter = " "
    THEN LET CONT = posición de carácter
    ELSE no hacer nada
  ENDIF
ENDLOOP
END
```

Ahora estamos en condiciones de codificar en lenguaje de programación a partir del seudolenguaje:

```
10 INPUT "INTRODUCIR NOMBRE COMPLETO";
   NOMBRE$
20 FOR L = 1 TO LEN (NOMBRE$)
30 LET CAR$ = MID$ (NOMBRE$,L,1)
40 IF CAR$ = " " THEN LET CONT = L
50 NEXT L
60 PRINT "ULTIMO ESPACIO ESTA EN POSICION";
   CONT
70 END
```

Nótese que la línea 10 es una entrada falsa, para verificar la rutina; la línea 60 es una salida falsa, también para verificación; y la línea 70 deberá ser cambiada por RETURN cuando la rutina se use como subrutina.

Y ahora el mismo proceso para la etapa cuarta:

```
2.º REFINAMIENTO
4. (Leer apellido)
BEGIN
Asignar a APELL$ los caracteres a la derecha del
último espacio de NOMBRE$
END
```

3.er REFINAMIENTO
```
4. (Leer apellido)
BEGIN
  READ NOMBRE$
  Localizar último espacio (llamar la subrutina
  ★ESPACIO★)
  LOOP mientras permanezcan caracteres en serie
  tras ESPACIO
    READ caracteres y asignar a APELL$
  ENDLOOP
END
```

Antes de continuar codificando en BASIC, se deben tener en cuenta algunos peligros potenciales. Al localizar el último espacio en el refinamiento anterior, el seudolenguaje exige el uso de la subrutina ★ESPACIO★, pero no sería posible escribirla en BASIC y comprobarla si ésta aún no se había escrito. Por regla general, no es práctico codificar cada módulo en BASIC (o cualquier otro lenguaje de alto nivel) hasta que no se haya desarrollado todo el programa en seudolenguaje. Sin embargo, si se desea comprobar un módulo, puede que resulte necesario escribir algunos valores de variables, entradas y salidas falsas. En el ejemplo anterior, CONT es la variable que tiene el valor de la posición del último espacio en NOMBRE$. Al comprobar, podemos hacer una pequeña trampa suponiendo que la rutina para realizar esto funciona correctamente:

```
10 LET NOMBRE$ = "PEDRO GONZALEZ"
20 LET CONT = 6
30 FOR L = CONT + 1 TO LEN (NOMBRE$)
40 LET APELL$ = APELL$ + MID$
   (NOMBRE$, L, 1)
50 NEXT L
60 PRINT "APELLIDO ES "; APELL$
70 END
```

A continuación se indica el proceso para encontrar el nombre de pila (etapa cinco). Recuérdese que se estableció que un nombre de pila es una concatenación de todos los caracteres alfabéticos hasta el último espacio del nombre completo. Los puntos, apóstrofos, espacios, etc., debían descartarse.

2.º REFINAMIENTO
```
5. (Leer nombre de pila)
BEGIN
LOOP hasta el último espacio mientras permanezcan
  caracteres en NOMBRE$
  Examinar caracteres
  IF carácter no es una letra
    THEN no hacer nada
    ELSE añadir carácter a NOMB$
  ENDIF
ENDLOOP
END
```

3.er REFINAMIENTO
```
5. (Leer nombre de pila)
BEGIN
LOOP hasta CONT mientras permanezcan caracteres
  LET CARTRAN$ = L° carácter en la variable
  IF CARTRAN$ no es una letra
    THEN no hacer nada
    ELSE LET NOMB$ = NOMB$ + CARTRAN$
  ENDIF
ENDLOOP
```

Ahora ya se puede codificar en BASIC, pero como se está en una etapa intermedia, vamos a utilizar sentencias en este lenguaje sin numerar, en un formato estructurado, de modo que se pueda comparar la estructura con la etapa anterior:

CODIFICACION
```
5. (Leer nombre de pila)
REM BEGIN
REM LOOP
  FOR L = 1 TO CONT -1
    LET CARTRAN$ = MID$ (NOMBRE$,L,1)
    LET CAR = ASC(CARTRAN$)
    IF CAR > 64 THEN NOMB$ =
      = NOMB$ + CHR$ (CAR)
    REM ENDIF
  NEXT L: REM ENDLOOP
REM END
```

En BASIC corriente esto tendría la forma siguiente:

```
10 FOR L = 1 TO CONT -1
20 LET CARTRAN$ = MID$ (NOMBRE$,L,1)
30 LET CAR = ASC(CARTRAN$)
40 IF CAR > 64 THEN NOMB$ = NOMB$ +
   + CHR$ (CAR)
50 NEXT L
60 END
```

Tal como está, sin embargo, este programa no funcionaría. Existen tres problemas: CONT requiere que se le asigne un valor; no está previsto introducir un nombre (asignando una variable a NOMBRE$); y no existe "salida" en forma de una frase impresa para que se pueda verificar si ha funcionado de forma correcta.

Si esta rutina formara parte de una subrutina, los parámetros pasados a ella (input) y los pasados desde ella (output) tendrían que ser tratados en otra parte del programa. Ésta es una consideración muy importante: el flujo de información en el interior de un programa debe ser siempre pensado con sumo cuidado antes de empezar a codificar en BASIC. Esto es particularmente importante cuando se emplean variables (CONT, por ejemplo) y se utiliza el mismo nombre de variable en diferentes partes del programa. De nada sirve llamar a una subrutina que emplea una variable como CONT si aquélla no tiene forma de saber cuál es el valor que se le atribuye. Si una subrutina indica el valor de CONT, éste seguirá siendo el mismo a menos que posteriormente se le asigne un nuevo valor, tal vez en otra subrutina. Ésta es una razón del porqué no es buena práctica en la programación dejar a medias un bucle, puesto que el valor de la variable del bucle será desconocido. Veamos a continuación las consecuencias de tener estos dos fragmentos del programa como partes de subrutinas diferentes en un programa:

Parte de subrutina X
```
FOR L = 1 TO LEN (PALABRA$)
LET CAR$ = MID$ (PALABRA$,L,1)
IF CAR$ = " . " THEN GOTO 1550
NEXT L
```

Parte de subrutina Y
```
FOR Q = 1 TO LIMIT
LET A(L) = P(Q)
NEXT Q
```

Esta parte de la subrutina Y está llevando valores a una variable que posee un subíndice determinado, el cual corresponde a la variable L. Si la subrutina Y

se utiliza después de la X, y la condición de la prueba en esta última ha sido encontrada (que uno de los caracteres sea un "."), el valor de L sería completamente impredecible y, por lo tanto, no se podría saber qué elemento de entre los valores de la variable le sería asignado en la subrutina Y. Aparte del error de ramificar un bucle, esta subrutina emplea también un GOTO, y esta práctica debiera evitarse. Los GOTO producen confusión y siempre que sea posible debería eludirse su utilización.

Para evitar confusión en el uso de variables, una buena norma es confeccionar una lista de ellas cuando se está en las etapas de seudolenguaje en el desarrollo del programa, junto con notas que indiquen para qué se están utilizando. Algunos lenguajes (pero no el BASIC) permiten establecer las variables como "locales" o "globales": esto es, tienen valores que se aplican a una parte del programa (locales) o a todo él (globales). Muchas variables, como, por ejemplo, las empleadas en bucles (la L en LET L = 1 TO 10), son casi siempre locales, por eso es muy conveniente dar el valor inicial de la variable antes de utilizarla (por ejemplo LET L = 0). Algunos lenguajes, como el PASCAL, hacen hincapié en esto; y aunque el BASIC siempre presupone que el valor inicial de una variable es 0 (mientras no se indique lo contrario), es recomendable efectuar lo anteriormente indicado.

Hasta ahora hemos formulado una definición razonable de un nombre, adecuada para la agenda computerizada que se intenta realizar, y se han desarrollado algunas rutinas que pueden manejar nombres en varias formas, que utilizaremos en nuestro programa completo. Ahora vamos a prescindir de los detalles de codificación del programa y entraremos a considerar la estructura de los "registros" en nuestro "archivo" de direcciones.

Los términos "registro" (*record*), "archivo" (*file*) y "campo" (*field*) tienen significados muy específicos y precisos en el mundo de la informática. Un *archivo* es un conjunto completo de información relacionada entre sí. En un sistema de ordenadores, sería un tema identificable, almacenado en un disco flexible o en una cinta de cassette y tendría su denominación propia, normalmente haciendo referencia al tema que trata. Podemos considerar la agenda completa como un archivo, y lo llamaremos AGENDA.

En un archivo tenemos *registros*, que son también conjuntos de información relacionada entre sí. Si imaginamos que el conjunto de direcciones constituye un fichero, el archivo sería la caja completa, llena de tarjetas, y cada ficha constituiría un registro: cada una de ellas con su nombre, dirección y número de teléfono.

En cada registro tenemos *campos*. Éstos pueden ser considerados como una o más filas de información afín contenida en un registro. Cada uno de los registros de nuestro archivo AGENDA constará de los siguientes campos: NOMBRE, DIRECCION y NUMERO TELEFONO. Un registro normal sería de la siguiente manera:

José Luis Padilla
General Dávila, 8
Madrid-3
Madrid
(91) 234 72 93

En este registro existen tres campos: el del nombre, que comprende letras alfabéticas (y probablemente el apóstrofo en ciertos nombres como Leopoldo O'Donnell, por ejemplo); el campo de la dirección, que consta de algunos números y bastantes letras; y el del número telefónico, que incluye únicamente números, sin tener en cuenta la eventualidad de que haya algún paréntesis en números como (91) 234 56 78. Antes de empezar a escribir un programa que maneje con flexibilidad información compleja como ésta, tenemos que decidir de qué manera representar los datos en el ordenador. Una forma podría ser considerar toda la información contenida en un registro como si fuera sólo una larga serie de caracteres. El problema que se presentaría con este tratamiento es que la extracción de una información específica sería extremadamente laboriosa. Supongamos por un momento que la siguiente entrada no es más que una larga serie de caracteres:

PERCIVAL R. BURTON
1056 AVENUE OF THE AMERICAS
RIO DEL MONTENEGRO
CALIFORNIA
U.S.A.
(1213) 884 5100

Si estuviéramos buscando los registros para hallar el número de teléfono de PERCIVAL R. BURTON, ¿sería correcto considerar que los últimos 15 caracteres del registro representan el número? ¿Qué sucedería si se hubiera incluido el prefijo de llamadas internacionales, de esta manera: 07 (1213) 884 5100? Entonces el número hubiera tenido un total de 18 caracteres. Para superar esta dificultad, se le asigna un campo separado, y el programa nos dará todos los caracteres (o números) de este campo cuando se le pida.

La dificultad de este planteamiento reside en que tiene que haber alguna forma de relacionar los diversos campos independientes, de modo que al referirnos a un campo (al del nombre, por ejemplo) nos pueda dar también los otros campos del registro. Una forma en que podría abordarse esto sería disponer de un campo suplementario, asociado con el registro únicamente con fines de clasificación. Si un registro fuera, por ejemplo, el decimoquinto del archivo, este campo de clasificación contendría el número 15. Esto podría utilizarse para indicar los elementos en un número de matrices. Para ilustrar este ejemplo, supongamos el registro siguiente:

Carmen Montero	campo NOMBRE
Balleneros, 12	campo CALLE
San Sebastián	campo CIUDAD
Guipúzcoa	campo PROVINCIA
(943) 45 72 73	campo NUMERO TELEFONO
017	campo CLASIFICACION

Si supiéramos el nombre de esta persona y quisiéramos conocer su número de teléfono, todo lo que tendríamos que hacer sería buscar el nombre de entre los elementos de la matriz que poseyeran esta información. Encontraríamos, entonces, que el elemento de la matriz en el cual estaba el nombre era el 17. Entonces lo único que faltaría por hacer sería encontrar el elemento decimoséptimo en la matriz NUMERO TELEFONO para obtener así el número de teléfono correcto.

Si tuviéramos varios amigos en la zona del Valle de Arán, es posible que un día quisiéramos utilizar el programa para buscar los datos de los que vivieran en la localidad de Viella, incluida en el campo CIUDAD. El programa, entonces, examinaría todos los campos CIUDAD y anotaría la localización de cada dato de Viella. Luego lo único que restaría por hacer, para que se imprimieran los nombres y direcciones de todos estos amigos, sería extraer todos los elementos que tengan el mismo número, pertenecientes a todas las matrices de cada registro "Viella". Al emplear este método, no sería necesario inspeccionar el campo CLASIFICACION. Esta técnica, además, cuenta con la ventaja de ser relativamente sencilla de realizar.

En el próximo capítulo de nuestro curso de programación estudiaremos algunos de los problemas que se presentan al buscar datos específicos a través de listas.

Ejercicio

■ Supongamos que unos registros que contengan los siguientes campos fueran adecuados para nuestra agenda computerizada:

campo NOMBRE
campo CALLE
campo CIUDAD
campo PROVINCIA
campo NUMERO TELEFONO

Supongamos que una de las opciones ofrecidas por un menú sea:

5. CREAR UNA NUEVA ENTRADA

Se teclea 5, y el programa se bifurca hacia la sección donde se han creado nuevos registros (se puede suponer que en la agenda ya no quedan entradas). Como el programa está completamente guiado por el menú, siempre se recibirán indicaciones sobre qué entradas hay que efectuar, con instrucciones tales como DAR ENTRADA AL NOMBRE, DAR ENTRADA A LA CALLE, DAR ENTRADA A LA CIUDAD, etc. A continuación se muestra el resultado que cabría esperar:

1. Un elemento en una matriz, para el nombre
2. Un elemento en una matriz, para la calle
3. Un elemento en una matriz, para la ciudad
4. Un elemento en una matriz, para la provincia
5. Un elemento en una matriz, para el teléfono

El problema consiste en desarrollar esto, mediante un proceso de programación *top-down* (de arriba abajo), utilizando un seudolenguaje hasta donde sea posible una conversión directa a BASIC. El seudolenguaje puede seguir las normas que usted le indique; sólo le sugerimos que emplee mayúsculas para palabras clave como IF, LOOP, etc., y minúsculas para descripciones en lenguaje corriente de las operaciones que deben realizarse. (Antes de comenzar el programa conviene examinar detenidamente el recuadro de "Complementos al BASIC".)

Complementos al BASIC



Paso 3

```
10 INPUT "DE ENTRADA AL NOMBRE
   COMPLETO";N$
15 LET CONT = 0
20 FOR L = 1 TO LEN N$
30 LET C$ = N$(L)
40 IF C$ = " " THEN LET CONT = L
50 NEXT L
60 PRINT "ULTIMO ESPACIO ESTA EN
   POSICION";CONT
70 STOP
```

En este proyecto de programa, las funciones en serie MID$, LEFT$ y RIGHT$ serán muy utilizadas. En el BASIC del Sinclair sustituir:

```
LEFT$(N$,N)   por  N$( TO N)
RIGHT$(N$,N)  por  N$(LEN(N$)-N+1 TO )
MID$(N$,P,N)  por  N$(P TO P+N-1)
MID$(N$,P,1)  por  N$(P)
```

Téngase en cuenta que los nombres de las variables alfanuméricas en el Spectrum no pueden estar formados por más de una letra (además de "$").

Paso 4

```
 5 LET A$ = " "
10 LET N$ = "PEDRO GONZALEZ"
20 LET CONT = 6
30 FOR L = CONT + 1 TO LEN N$
40 LET A$ = A$ + N$(L)
50 NEXT L
60 PRINT "APELLIDO ES ";A$
70 STOP
```

Paso 5

```
 5 LET C$ = " "
10 FOR L = 1 TO CONT - 1
20 LET T$ = N$(L)
30 LET CAR = CODE T$
40 IF CAR > 64 THEN LET C$ =
   = C$ + CHR$ CAR
50 NEXT L
60 STOP
```

En este fragmento ha surgido el problema que acarrea el designar las variables alfanuméricas con una sola letra: N$ es el equivalente, en el Spectrum, a la variable NOMBRE$; por lo tanto, C$ debe corresponder a la variable NOMB$.

Parte de subrutina X

```
FOR L = 1 TO LEN P$
LET C$ = P$(L)
IF C$ = "." THEN GOTO 1550
NEXT L
```

Parte de subrutina Y

```
FOR Q = 1 TO LIMIT
LET A(L) = P(Q)
NEXT Q
```



Entre los ordenadores personales más populares, sólo el BBC Micro admite nombres de variables largos, tal como NOMBRE$. El Spectrum permite usar nombres de variables numéricas largos, pero únicamente una letra para nombres de variables alfanuméricas. El Dragon 32, Vic-20 y Commodore 64 pueden utilizar nombres de variables largos, pero sólo los dos caracteres primeros son significativos, por tanto usar NOMBRE$ es válido, pero hará referencia a la misma posición de memoria que NOMB$: tienen iguales los dos primeros caracteres. En el Oric-1, los nombres de las variables no pueden incluir más de dos caracteres (primero una letra, luego un número o una letra), mientras que el Lynx admite sólo una letra, si bien se pueden emplear mayúsculas y minúsculas con significado distinto.

Papel de calco

Las figuras dibujadas en papel pueden trasladarse al ordenador mediante un digitalizador o un tablero de gráficos

Una de las características más importantes de la actual generación de ordenadores personales es su capacidad de crear gráficos. Con unas sencillas instrucciones se pueden crear dibujos y formas y cambiar colores. Todo ello requiere tener conocimientos de programación, puesto que aún no es posible crear una imagen sobre el papel y luego cargarla en el ordenador. Los lápices ópticos (véase p. 156) facilitan la edición y manipulación de una imagen una vez ésta aparece en la pantalla, pero no se pueden utilizar para copiar una figura a partir de una hoja de papel.

Los proyectistas de coches, aeroplanos y microprocesadores, así como los decoradores de interiores, arquitectos de parques y creadores de moda pueden beneficiarse del sistema de gráficos de un ordenador. Una vez que el diseño se ha almacenado de forma segura en la memoria del ordenador, puede intentarse la realización de adiciones y alteraciones sin echar a perder materias primas valiosas. Por lo tanto, será necesario disponer de un dispositivo de entrada que sea capaz de trasladar las líneas y curvas del dibujo o diseño a un lenguaje que pueda ser entendido por el ordenador.

En el mercado profesional, el "tablero de gráficos" se ha estado empleando casi tanto como el ordenador. Sin embargo, sólo desde hace muy poco tiempo es asequible para el usuario del ordenador personal. Los tableros gráficos de alta precisión, también conocidos como "digitalizadores", emplean una vasta gama de técnicas para producir la información requerida. Los sistemas más exactos pueden proporcionar una resolución de imagen de alrededor de 1/4 mm, suficiente para ser usada por ingenieros y delineantes.

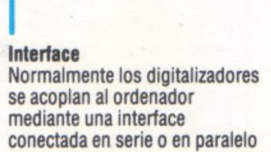
Todos los digitalizadores constan de una base plana sobre la que se apoya el papel en el cual se ha dibujado o pintado. Un cursor, que puede ser una pluma corriente o un dispositivo electrónico sofisticado, es luego desplazado sobre la imagen. La posición del cursor es detectada por el digitalizador y transmitida al ordenador en forma de un cambiante par de coordenadas.

Los dos sistemas más precisos —magnético y capacitivo— funcionan mediante una trama de cables dispuestos en forma de rejilla en la parte inferior del tablero.

En el sistema magnético, el cursor consiste en una pequeña lupa con una alidada transversal que se desplaza sobre la imagen. Alrededor de la lupa existe un solenoide que transmite una señal de baja potencia y alta frecuencia. La señal es detectada por la rejilla y suministra una medida directa a la posición del cursor.

El sistema capacitivo trabaja en la forma inversa; se suministra a la rejilla una serie de pulsos codificados y el cursor recoge la señal.

Una forma alternativa la constituye el sistema acústico. El cursor está cargado electrostáticamente, y al entrar en contacto con el tablero se produce una minúscula chispa. El tiempo que necesita la onda acústica creada por la chispa para alcanzar dos micrófonos proporciona una medida de la posición del cursor. Entre otras características, este sistema ofrece la posibilidad de digitalizar en tres dimensiones por medio de una señal que atraviese el objeto.



Tablero base
En esta superficie se sitúa la imagen a digitalizar. En algunos sistemas, se aplica al tablero una carga electrostática para que el papel quede completamente plano. Es muy importante que la imagen no se desplace con relación al tablero

En el extremo inferior de la escala, desde el punto de vista de la precisión, se encuentra el tablero sensible a la presión: en él la figura es contorneada por un cursor. Dos hojas conductoras de electricidad están separadas por un aislante celular; a estas dos capas se las alimenta con dos señales diferentes de alta frecuencia. La señal detectada por el cursor cuando éste efectúa una conexión eléctrica entre las dos hojas, suministra una medida de su posición. Un problema típico que se encuentra en esta clase de sistemas es el que se refiere a los cambios en la resistencia de la superficie, debidos a su mal estado o a las diferencias de presión producidas por la mano. Dada la limitada resolución de los gráficos de los ordenadores personales, la precisión de este método resulta más que adecuada para las máquinas de hoy en día.


Simon Lewis

Trazado de mapas
Uno de los usos profesionales más comunes para los digitalizadores es la recogida de datos de mapas y reconocimiento de terrenos. En la figura, el ordenador se utiliza para localizar campos petrolíferos a partir de datos geológicos digitalizados

Tablero de procesamiento
Contiene un microprocesador, así como ROM y RAM. Por ello puede ofrecer al ordenador información en forma de pares de coordenadas X-Y

Rejilla receptora
Está adosada en la parte inferior del tablero, y puede recoger la señal emitida por la bobina. La anchura de la malla es considerablemente más amplia que la correspondiente a la alta resolución del digitalizador, debido a que el sistema de circuitos de procesamiento es capaz de interpolar la potencia relativa de la señal recogida por los cables de la rejilla

Los digitalizadores más baratos y sencillos son los pantógrafos, basados en el anticuado sistema de dibujo formado por dos brazos articulados. Los pantógrafos utilizan valores de coordenadas para proporcionar una medida directa de la posición del cursor. Unas resistencias variables, montadas en las dos uniones, suministran voltajes proporcionales a los ángulos del "hombro" y el "codo" del brazo articulado. La resolución del pantógrafo está limitada por la precisión tanto de las resistencias variables como de las articulaciones mecánicas; el valor normal de ella suele ser sólo de alrededor de un cinco por ciento. Sin embargo, los pantógrafos más complejos, basados en mediciones ópticas de la rotación de las uniones, pueden ofrecer unos resultados mucho mejores, si bien quedan todavía muy por debajo de las posibilidades que ofrecen los sistemas magnético y capacitivo.

Los tableros ópticos usan un entramado de haces infrarrojos para detectar la posición del cursor. No son tan sensibles como los otros sistemas, pero son bastante adecuados como para permitir utilizar un dedo para seleccionar un tema de un menú de programas. En algunas aplicaciones la fuente de los rayos infrarrojos y los detectores están situados alrededor del borde de la unidad de representación visual, suministrando una pantalla verdaderamente interactiva, sobre la cual se pueden dibujar figuras simplemente con el movimiento de un dedo.

Los datos reales producidos por un tablero de gráficos o un digitalizador deben ser convertidos en información adecuada para ser representada en la pantalla, y con este fin la mayoría de los productos comerciales disponen del software necesario. Sin embargo, la utilidad de los tableros gráficos va más allá de la mera introducción de datos. Una vez la información ha sido almacenada en el ordenador, el tablero puede usarse como herramienta de montaje, permitiendo añadir o cambiar colores y modificar formas. La superficie del tablero puede ser programada para actuar como un menú que selecciona opciones estándar de un programa, de forma que sólo sea necesario usar el teclado para seleccionar las funciones principales. Los sistemas de animación con ayuda de ordenador (véase p. 181) tienen un tablero de gráficos de alta calidad como componente principal de su dispositivo de entrada.

Gottfried Leibniz

1646
Nace el 1 de julio, en Leipzig

1661
Se matricula en la Universidad de Leipzig y se gradúa a los 17 años

1661-1670
Trabaja como abogado y diplomático. Publica un ensayo sobre "El arte de la combinación"

1672
Desarrolla en París el principio de la "razón suficiente"

1673
Presenta en la Royal Society la máquina calculadora

1675
Realiza estudios sobre la gravedad independientemente de Newton

1676
Aportación a la dinámica mediante el concepto de energía cinética

1678
Es nombrado bibliotecario y consejero del duque de Hannover

1679
Desarrolla la teoría del sistema binario

1683
Publica el panfleto "El más cristiano dios de la guerra", un ataque a Luis XIV

1690-1699
Su estudio genealógico de la Casa de Hannover se amplía a una Historia del mundo

1700
Organiza la Academia de las Ciencias de Berlín

1714
Se le designa para establecer el derecho de sucesión de Jorge I al trono vacante de Inglaterra

1716
Muere el 14 de noviembre, en Hannover


Cortesía Science Museum

Los científicos que trabajan en la creación de la quinta generación de ordenadores muestran cada vez mayor interés en las investigaciones de este pensador

Gottfried Wilhelm Leibniz fue la figura científica principal de su tiempo —el período conocido como el Siglo de las Luces—. Nació en la ciudad centroeuropea de Leipzig en 1646, y murió en Hannover en 1716. Durante sus setenta años de vida (el tipo de número exacto que se puede esperar de un matemático), formuló los principios del cálculo infinitesimal, realizó estudios sobre la dinámica y contribuyó con valiosas aportaciones a los campos de la geología, teología, historia, lingüística y filosofía. Y lo más importante para nosotros: desarrolló teorías básicas para la creación del ordenador.

Empezó a viajar a la edad de veinte años, después de que la Universidad de Leipzig rehusara concederle el doctorado en leyes debido a su juventud. Carente de recursos propios, Leibniz se vio obligado a desempeñar trabajos que interferían con sus investigaciones científicas. Con poco más de veinte años fue abogado y diplomático; más tarde fue bibliotecario y consejero de la familia real.

Los intereses de Leibniz eran muy amplios y su naturaleza cosmopolita le condujo a viajar frecuentemente por Europa, manteniendo conversaciones y contactos con todos los grandes pensadores de su tiempo. Era un prolífico escritor de cartas: mantuvo correspondencia con más de 600 personas.

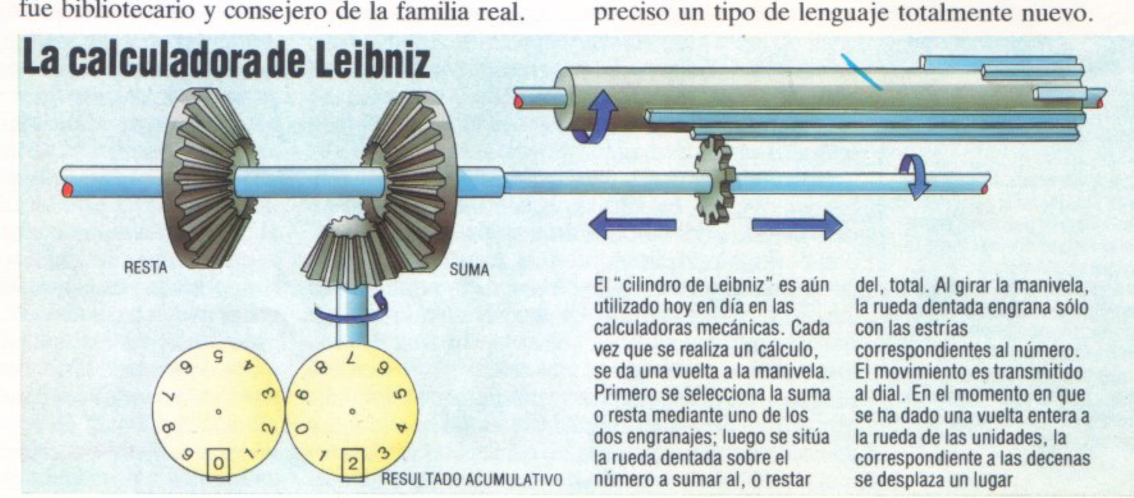
Su primera contribución importante a la filosofía se produjo en 1672, cuando formuló el principio de la "razón suficiente". Éste sostenía, simplemente, que tiene que haber una razón para todo, y "todo conduce al bien en el mejor de los mundos".

Centrando su interés en las matemáticas, empezó a trabajar en el perfeccionamiento de la máquina de sumar de Blaise Pascal, inventada en 1642 (véase p. 86). Leibniz intentó mejorarla de forma que fuera capaz de multiplicar y dividir. Lo logró mediante un dispositivo mecánico llamado *cilindro de Leibniz* (véase la ilustración en la parte inferior de la página). El mecanismo de Leibniz fue un adelanto decisivo para su tiempo. Antes, debido a la complejidad de multiplicar con números romanos, esta operación aritmética se enseñaba sólo en los centros de estudio de mayor categoría. Una máquina que pudiera multiplicar mecánicamente la volvía más accesible. Después de haber perfeccionado esta máquina, Leibniz centró sus esfuerzos en la creación de un método que permitiera convertir el sistema decimal en otro de base binaria.

La mayor ambición del filósofo y matemático alemán era idear un lenguaje universal que pudiera emplear la claridad y precisión de las matemáticas para resolver cualquier problema que se le pudiera presentar a la humanidad. Su lenguaje debería utilizar símbolos abstractos para representar los "átomos" fundamentales del entendimiento, con un conjunto de reglas para manejar estos símbolos. Su intento resultó fallido; pero sus ideas fueron recogidas a principios del siglo XX por Bertrand Russell, quien intentó explicar las matemáticas en términos de "lenguaje" lógico formal.

En los últimos años, se ha visto incrementado el interés en las investigaciones de Leibniz por parte de los científicos que trabajan en el proyecto a largo plazo de crear la quinta generación de ordenadores. Estas máquinas, según se cree, podrán resolver cualquier problema humano con la misma velocidad y fiabilidad con que los ordenadores actuales ejecutan cálculos matemáticos. Para ello será preciso un tipo de lenguaje totalmente nuevo.

La calculadora de Leibniz



El "cilindro de Leibniz" es aún utilizado hoy en día en las calculadoras mecánicas. Cada vez que se realiza un cálculo, se da una vuelta a la manivela. Primero se selecciona la suma o resta mediante uno de los dos engranajes; luego se sitúa la rueda dentada sobre el número a sumar al, o restar del, total. Al girar la manivela, la rueda dentada engrana sólo con las estrías correspondientes al número. El movimiento es transmitido al dial. En el momento en que se ha dado una vuelta entera a la rueda de las unidades, la correspondiente a las decenas se desplaza un lugar

Kevin Jones

DONDE CONSEGUIR TU sinclair

ALAVA
COMPONENTES ELECTRONICOS GAZTEIZ
Domingo Beltrán, 58 (Vitoria)
DEL CAZ
Avda. Gazteiz, 58 (Vitoria)
VALBUENA
Virgen Blanca, 1 (Vitoria)

ALBACETE
ELECTRO MIGUEL
Tesifonte Gallego, 27
TECON
Maria Marin, 13

ALICANTE
ASEMCA (Villena)
Avda. de la Constitución, 54 (Villena)
CONSULTING DESARROLLO INFORMATICO
Pais Valencia, 54 (Alcoy)
COMPONENTES ELECTRONICOS LASER
Jaime M.ª Buch, 7
ELECTRODATA LEVANTE
San Vicente, 28
ELECTRONICA AITANA
Limones, s/n. Edificio Urgull (Benidorm)
ELECTRONICA OHMIO
Avda. El Hamed, 1
LIBRERIA LLORENS
Alameda, 50 (Alcoy)

AVILA
FELIX ALONSO
San Segundo, 15

BADAJOZ
MECANIZACION EXTREMEÑA
Vicente Barantes, 18
SONYTEL
Villanueva, 16

BARCELONA
ARTO
C/ Angli, 43
BERENGUERAS
C/ Diputación, 219
CATALANA D'ORDINADORS
C/ Trafalgar, 70
CECSA
C/ Mallorca, 367
COMPUTERLAND
C/ Infanta Carlota, 89
COMPUTERLAND
Trav. de Dalt, 4
COPIADUX
C/ Dos de Mayo, 234
D. P. 2000
C/ Sabino de Arana, 22-24
DIOTRONIC
C/ Conde Borrell, 108
EL CORTE INGLES
Avda. Diagonal, 617-619
EL CORTE INGLES
Pza. Cataluña, 14
ELECTRONICA H. S.
C/ S. José Oriol, 9
ELECTRONICA SAUQUET
C/ Guilleries, 10
ELEKTROCOMPUTER
Vía Augusta, 120
EXPOCOM
C/ Villarroel, 68
GUIBERNAU
C/ Sepúlveda, 104
INSTA-DATA
P.º S. Juan, 115
MAGIAL
C/ Sicilia, 253
MANUEL SANCHEZ
Pza. Major, 40 (Vic)
MILLIWATTS
C/ Meléndez, 55 (Mataró)
ONDA RADIO
Gran Via, 581
RADIO ARGANY
C/ Borrell, 45
RADIO SONDA
Avda. Abad Margat, 77 (Tarrasa)
RAMEL ELECTRONICA
Cr. de Vic, 3 (Manresa)
REDISA GESTION
Avda. Sarriá, 52-54
RIFE ELECTRONICA
C/ Aribau, 80, 5.º, 1.ª
SERVICIOS ELECTRONICOS VALLES
Pza. del Gas, 7 (Sabadell)
SISTEMA
C/ Balmes, 434
S. E. SOLE
C/ Muntaner, 10
SUMINISTROS VALLPARADIS
C/ Dr. Ferrén, 172 (Tarrasa)
TECNOHIFI, S. A.
C/ La Rambleta, 19
VIDEOCOMPUT
P.º Pep Ventura, 9, Bl. C, Bjos. Bis (Vic)

BURGOS
COMELECTRIC
Calzada, 7
ELECTROSON
Conde don Sancho, 6

CACERES
ECO CACERES
Diego Maria Crehuet, 10-12

CADIZ
ALMACENES MARISOL
Camoens, 11 (Ceuta)
INFORSA
Avda. Fuerzas Armadas, 1 (Algeciras)
ELECTRONICA VALMAR
Ciudad de Santander, 8
M. R. CONSULTORES
Multi Centro Merca 80 (Jerez de la Frontera)
PEDRO VAREA
Porvera, 36 (Jerez de la Frontera)
LEO COMPUTER
Garcia Escamez, 3
SONYTEL
Queipo de LLano, 17
SONYTEL
José Luis Diez, 7
T L C Y AUTOMATICA
Dr. Herrera Quevedo, 2

CASTELLON
NOÙ DESPACH'S
Rey D. Jaime, 74

CIUDAD REAL
COMERCIAL R. P.
Travesera de Coso, 2 (Valdepeñas)
ECO CIUDAD REAL
Calatrava, 8

CORDOBA
ANDALUZA DE ELECTRONICA
Felipe II, 15
CONTROL
Conde de Torres Cabrera, 9
ELECTRONICA PADILLA
Sevilla, 9
MORM
Plaza Colón, 13
SONYTEL
Arte, 3
Avda. de los Mozárabes, 7

CUENCA
SONYTEL
Dalmacio Garcia Izcara, 4

GERONA
AUDIFILM
C/ Albareda, 15
CENTRE DE CALCUL DE CATALUNYA
C/ Barcelona, 35
S. E. SOLE
C/ Sta. Eugenia, 59

GRANADA
INFORMATICA Y ELECTRONICA
Melchor Almagro, 8
SONYTEL
Manuel de Falla, 3
TECNIGAR
Ancha de Gracia, 11

GRANOLLERS
COMERCIAL CLAPERA
C/ Maria Maspons, 4

GUIPUZCOA
ANGEL IGLESIAS
Sancho el Sabio, 7-9
BHP NORTE
Ramón M.ª Lili, 9
ELECTROBON
Reina Regente, 4

HUELVA
SONYTEL
Ruiz de Alda, 3

HUESCA
ELECTRONICA BARREU
M.ª Auxiliadora, 1

IBIZA
IBITEC
C/ Aragón, 76

JAEN
CARMELO MILLA
Coca de la Piñera, 3
MARA ILUMINACION
Avda. Linares, 13 (Ubeda)
MICROJISA
Garcia Rebull, 8
SONYTEL
José Luis Diez, 7
SONYTEL
Pasaje del Generalisimo, 3 (Linares)

LA CORUÑA
DAVIÑA
República de El Salvador, 29 (Santiago)
PHOTOCOPY
Teresa Herrera, 9
SONYTEL
Avda. de Arteijo, 4
SONYTEL
Tierra, 37

LAS PALMAS DE GRAN CANARIA
COMPUTERLAND
Carvajal, 4
CHANRAI
Triana, 3
EL CORTE INGLES
José Mesa y López, 18

LEON
ELECTROSON
Avda. de la Facultad, 15
MICRO BIERZO
Carlos I, 2 (Ponferrada)
RADIO RACE
Modesto Lafuente, 3

LERIDA
SELEC
C/ Ferrer y Busquet, 14 (Mollerusa)
SEMIC
C/ Pi y Margall, 47

LUGO
ELECTROSON
Concepción Arenal, 38
SONYTEL
Primo de Rivera, 30

MADRID
ALFAMICRO
Augusto Figueroa, 16
BELLTON'S
Torpedero Tucumán, 8
CHIPS-TIPS
Pto. Rico, 21
CMP
Pto. Santa Maria, 128
COMPUTERLAND
Castelló, 89
COSESA
Barquillo, 25
DINSA
Gaztambide, 4
DISTRIBUIDORA MADRILEÑA
Todos sus centros
ELECTROSON
Duque de Sexto, 15 (y otros centros)
INVESMICROSTORE
Génova, 7
J.P. MICROCOMPUT
Montesa, 44
EL CORTE INGLES
Todos sus centros
ELECTRONICA SANDOVAL
Sandoval, 4
PENTA
Dr. Cortezo, 12
RADIO CINEMA
Antonio Acuña, 3
RADIO QUER
Todos sus centros
SONYTEL
Clara del Rey, 24 (y todos sus centros)
SONICAR
Vallehermoso, 19
VIDEOMUSICA
Orense, 28

MALAGA
EL CORTE INGLES
Prolongación Alameda, s/n.
INGESCON
Edificio Galaxia
SONYTEL
Salitre, 13

MELILLA
OFI-TRONIC
Hermanos Cayuela, 11

MENORCA
ELECTRONICA MENORCA
C/ Miguel de Veri, 50 (Mahon)

MURCIA
COMPUTER LIFE
Alameda San Antón, 2 (Cartagena)
EL CORTE INGLES
Libertad, 1
ELECTRONICA COMERCIAL CRUZ
Rio Segura, 2
MICROIN
Gran Via, 8

NAVARRA
ENER
Paulino Caballero, 39
GABINETE TECNICO EMPRESARIAL
Juan de Labrit, 3
JOSE LUIS DE MIGUEL
Arrieta, 11 bis

OVIEDO
AUTECA
Valentin Masip, 25
EDIMAR
Cangas de Onis, 4-6 (Gijón)
ELECTRONICA RATO
Versalles, 45 (Avilés)
RADIO NORTE
Uria, 20
RESAM ELECTRONICA
San Agustin, 12 (Gijón)
RETELCO
Cabrales, 31 (Gijón)
SELECTRONIC
Fermin Canellas, 3

ORENSE
SONYTEL
Concejo, 11

PONTEVEDRA
EL CORTE INGLES
Gran Via, 25 (Vigo)
ELECTROSON
Santa Clara, 32
ELECTROSON
Venezuela, 32 (Vigo)
SONYTEL
Salvador Moreno, 27
SONYTEL
Gran Via, 52 (Vigo)
TEFASA COMERCIAL
San Salvador, 4 (Vigo)

PALMA DE MALLORCA
GILFT
Via Alemania, s/n
IAM
C/ Cecilio Metlo, 5
TRON INFORMATICA
C/ Juan Alcover, 54, 6.º C

LA RIOJA
YUS COMESSA
Cigüeña, 15

SALAMANCA
DEL AMO
Arco, 5
PRODISTELE
España, 65

SANTANDER
LAINZ S. A.
Reina Victoria, 127
RADIO MARTINEZ
Dr. Jiménez Diaz, 13

SEGOVIA
ELECTRONICA TORIBIO
Obispo Quesada, 8

SEVILLA
A.D.P.
San Vicente, 3
EL CORTE INGLES
Duque de la Victoria, 10
SCI
Aceituno, 8
SONYTEL
Pages del Corro, 173
Adriano, 32

TARRAGONA
AIA
Rambla Nova, 45, 1.º
CIAL. INFORMATICA TARRAGONA
C/ Gasómetro, 20
ELECTRONICA REUS
Avda. Prat de la Riba, 5 (Reus)
SEIA
Rambla Vella, 7 B
S.E. SOLE
C/ Cronista Sese, 3
T. V. HUGUET
Pza. Major, 14 (Montblac)
VIRGILI
C/ Dr. Gimbernat, 19 (Reus)

STA. CRUZ DE TENERIFE
COMPUTERLAND
Mendez Núñez, 104 B
TRENT CANARIAS
Serrano, 41

VALENCIA
ADISA
San Vicente, 33 (Gandia)
CESPEDES
San Jacinto, 6
COMPUTERLAND
Marqués del Turia, 53
DIRAC
Blasco Ibáñez, 116
EL CORTE INGLES
Pintor Sorolla, 26
Meléndez Pidal, 15
PROMOCION INFORMATICA
Pintor Zariñena, 12

VALLADOLID
SONYTEL
León, 4

VIZCAYA
BILBOMICRO
Aureliano del Valle, 7
DATA SISTEMAS
Henao, 58
DISTRIBUIDORA COM
Gran Via, 19-21 y todos sus centros
EL CORTE INGLES
Gran Via, 9
ELECTROSON
Alameda de Urquijo, 71
San Vicente, 18 (Baracaldo)
GESCO INFORMATICA
Alameda de Recalde, 76
KEYTRON
Hurtado de Amezaga, 20

ZAMORA
MEZZASA
Victor Gallego, 17

ZARAGOZA
EL CORTE INGLES
Sagasta, 3
SONYTEL
Via Pignatelli, 29-31



DISTRIBUIDOR EXCLUSIVO:
INVESTRONICA

Central Comercial
TOMAS BRETON, 60
TELF. 468 03 00
TELEX 23399 IYCO E
MADRID

Delegación Cataluña
MUNTANER, 565
TELF. 212 68 00
BARCELONA


10013
9 788485 822836